COLA: Consistent Learning with Opponent-Learning Awareness

Timon Willi [* 1]   Alistair Letcher [*]   Johannes Treutlein [* 2 3]   Jakob Foerster [1]

Abstract

Learning in general-sum games is unstable and frequently leads to socially undesirable (Pareto-dominated) outcomes. To mitigate this, Learning with Opponent-Learning Awareness (LOLA) introduced *opponent shaping* to this setting, by accounting for each agent's influence on their opponents' anticipated learning steps. However, the original LOLA formulation (and follow-up work) is *inconsistent* because LOLA models other agents as *naive learners* rather than LOLA agents. In previous work, this inconsistency was suggested as a cause of LOLA's failure to preserve stable fixed points (SFPs). First, we formalize consistency and show that higher-order LOLA (HOLA) solves LOLA's inconsistency problem if it converges. Second, we *correct a claim* made in the literature by Schäfer and Anandkumar (2019), proving that Competitive Gradient Descent (CGD) does *not* recover HOLA as a series expansion (and fails to solve the consistency problem). Third, we propose a new method called Consistent LOLA (COLA), which *learns* update functions that are consistent under mutual opponent shaping. It requires no more than second-order derivatives and learns consistent update functions even when HOLA fails to converge. However, we also prove that even consistent update functions do not preserve SFPs, contradicting the hypothesis that this shortcoming is caused by LOLA's inconsistency. Finally, in an empirical evaluation on a set of general-sum games, we find that COLA finds prosocial solutions and that it converges under a wider range of learning rates than HOLA and LOLA. We support the latter finding with a theoretical result for a simple game.

[*]Equal contribution [1]Department of Engineering Science, University of Oxford, United Kingdom [2]Department of Computer Science, University of Toronto, Canada [3]Vector Institute, Toronto, Canada. Correspondence to: <timon.willi@eng.ox.ac.uk>.

1. Introduction

Much research in deep multi-agent reinforcement learning (MARL) has focused on zero-sum games like Starcraft and Go (Silver et al., 2017; Vinyals et al., 2019) or fully cooperative settings (Oroojlooyjadid & Hajinezhad, 2019). However, many real-world problems, e.g. self-driving cars, contain both cooperative and competitive elements, and are thus better modeled as general-sum games. One such game is the famous Prisoner's Dilemma (Axelrod & Hamilton, 1981), in which agents have an individual incentive to defect against their opponent, even though they would prefer the outcome in which both cooperate to the one where both defect. A strategy for the infinitely iterated version of the game (IPD) is tit-for-tat, which starts out cooperating and otherwise mirrors the opponent's last move. It achieves mutual cooperation when chosen by both players and has proven to be successful at IPD tournaments (Axelrod & Hamilton, 1981). If MARL algorithms are deployed in the real world, it is essential that they are able to cooperate with others and entice others to cooperate with them, using strategies such as tit-for-tat (Dafoe et al., 2021). However, naive gradient descent and other more sophisticated methods (Korpelevich, 1977; Mescheder et al., 2017; Balduzzi et al., 2018; Mazumdar et al., 2019; Schäfer & Anandkumar, 2019) converge to the mutual defection policy under random initialization (Letcher et al., 2019b).

An effective paradigm to improve learning in general-sum games is *opponent shaping*, where agents take into account their influence on the anticipated learning step of the other agents. LOLA (Foerster et al., 2018a) was the first work to make explicit use of opponent shaping and is one of the only general learning methods designed for general-sum games that obtains mutual cooperation with the tit-for-tat strategy in the IPD. While LOLA discovers these prosocial equilibria, the original LOLA formulation is inconsistent because LOLA agents assume that their opponent is a *naive learner*. This assumption is clearly violated if two LOLA agents learn together. It has been suggested that this inconsistency is the cause for LOLA's main shortcoming, which is not maintaining the stable fixed points (SFPs) of the underlying game, even in some simple quadratic games (Letcher 2018, pp. 2, 26; see also Letcher et al. 2019b).

Contributions. To address LOLA's inconsistency, we first revisit the concept of *higher-order* LOLA (HOLA) (Foerster et al., 2018a) in Section 4.1. For example, *second-order* LOLA assumes that the opponent is a *first-order* LOLA agent (which in turn assumes the opponent is a naive learner) and so on. Supposing that HOLA converges with increasing order, we define *infinite-order* LOLA (iLOLA) as the limit. Intuitively, two iLOLA agents have a *consistent* view of each other since they accurately account for the learning behavior of the opponent under mutual opponent shaping. Based on this idea, we introduce a formal definition of *consistency* and prove that, if it exists, iLOLA is indeed consistent (Proposition 4.3).

Second, in Section 4.2, we correct a claim made in previous literature, which would have provided a closed-form solution of the iLOLA update. According to Schäfer & Anandkumar (2019), the Competitive Gradient Descent (CGD) algorithm recovers HOLA as a series expansion. If true, this would imply that CGD coincides with iLOLA, thus solving LOLA's inconsistency problem. We prove that this is untrue: CGD's series expansion does, in general, not recover HOLA, CGD does not correspond to iLOLA, and CGD does not solve the inconsistency problem (Proposition 4.4).

In lieu of a closed-form solution, a naive way of computing the iLOLA update is to iteratively compute higher orders of LOLA until convergence. However, there are two main problems with addressing consistency using a limiting update: the process may diverge and typically requires arbitrarily high derivatives. To address these, in Section 4.3, we propose Consistent LOLA (COLA) as a more robust and efficient alternative. COLA learns a pair of consistent update functions by explicitly minimizing a differentiable measure of consistency inspired by our formal definition. We use the representational power of neural networks and gradient based optimization to minimize this loss, resulting in *learned* update functions that are mutually consistent. By reframing the problem as such, we only require up to second-order derivatives.

In Section 4.4, we prove initial results about COLA. First, we show that COLA's solutions are not necessarily unique. Second, despite being consistent, COLA does not recover SFPs, contradicting the prior belief that this shortcoming is caused by inconsistency. Third, to show the benefit of additional consistency, we prove that COLA converges under a wider range of look-ahead rates than LOLA in a simple general-sum game.

Finally, in Sections 5 and 6, we report our experimental setup and results, investigating COLA and HOLA and comparing COLA to LOLA and CGD in a range of games. We experimentally confirm our theoretical result that CGD does not equal iLOLA. Moreover, we show that COLA converges under a wider range of look-ahead rates than HOLA and

LOLA, and that it is generally able to find socially desirable solutions. It is the only algorithm consistently converging to the fair solution in the Ultimatum game, and while it does not find tit-for-tat in the IPD (unlike LOLA), it does learn policies with near-optimal total payoff. We find that COLA learns consistent update functions even when HOLA diverges with higher order and its updates are similar to iLOLA when HOLA converges. Although COLA solutions are not unique in theory, COLA empirically tends to find similar solutions over different runs.

2. Related work

General-sum learning algorithms have been investigated from different perspectives in the reinforcement learning, game theory, and GAN literature (Schmidhuber, 1991; Barto & Mahadevan, 2003; Goodfellow et al., 2014; Racanière et al., 2017). Next, we will highlight a few of the approaches to the mutual opponent shaping problem.

Opponent modeling maintains an explicit belief of the opponent, allowing to reason over their strategies and compute optimal responses. Opponent modeling can be divided into different subcategories: There are classification methods, classifying the opponents into pre-defined types (Weber & Mateas, 2009; Synnaeve & Bessière, 2011), or policy reconstruction methods, where we explicitly predict the actions of the opponent (Mealing & Shapiro, 2017). Most closely related to opponent shaping is recursive reasoning, where methods model nested beliefs of the opponents (He & Boyd-Graber, 2016; Albrecht & Stone, 2017; Wen et al., 2019).

In comparison, COLA assumes that we have access to the ground-truth model of the opponent, e.g., the opponent's payoff function, parameters, and gradients, putting COLA into the framework of differentiable games (Balduzzi et al., 2018). Various methods have been proposed, investigating the local convergence properties to different solution concepts (Mescheder et al., 2017; Mazumdar et al., 2019; Letcher et al., 2019b; Schäfer & Anandkumar, 2019; Azizian et al., 2020; Schäfer et al., 2020; Hutter, 2021). Most of the work in differentiable games has not focused on opponent shaping or consistency. Mescheder et al. (2017) and Mazumdar et al. (2019) focus solely on zero-sum games without shaping. To improve upon LOLA, Letcher et al. (2019b) suggested Stable Opponent Shaping (SOS), which applies ad-hoc corrections to the LOLA update, leading to theoretically guaranteed convergence to SFPs. However, despite its desirable convergence properties, SOS still does not solve the conceptual issue of inconsistent assumptions about the opponent. CGD (Schäfer & Anandkumar, 2019) addresses the inconsistency issue for zero-sum games but not for general-sum games. The exact difference between CGD, LOLA and our method is addressed in Section 4.2. Model-Free Opponent Shaping (M-FOS) (Lu et al., 2022)

Table 1. On the Tandem game: (a) Log of the squared consistency loss, where e.g. HOLA6 is sixth-order higher-LOLA. (b) Cosine similarity between COLA and LOLA, HOLA3, and HOLA6 over different look-ahead rates. The values represent the mean of a 1,000 samples, uniformly sampled from the parameter space Θ. Error bars represent one standard deviation over 10 COLA training runs.

<table>
<tr><td colspan="5" align="center">(a)</td><td colspan="4" align="center">(b)</td></tr>
<tr><td>α</td><td>LOLA</td><td>HOLA3</td><td>HOLA6</td><td>COLA</td><td>α</td><td>LOLA</td><td>HOLA3</td><td>HOLA6</td></tr>
<tr><td>1.0</td><td>128.0</td><td>512</td><td>131072</td><td>3e-14±2e-15</td><td>1.0</td><td>0.94±0.04</td><td>0.94±0.04</td><td>0.94±0.04</td></tr>
<tr><td>0.5</td><td>12.81</td><td>14.05</td><td>12.35</td><td>2e-14±5e-15</td><td>0.5</td><td>0.88±0.12</td><td>0.88±0.12</td><td>0.08±0.13</td></tr>
<tr><td>0.3</td><td>2.61</td><td>2.05</td><td>0.66</td><td>4e-14±3e-15</td><td>0.3</td><td>0.92±0.01</td><td>0.91±0.01</td><td>0.80±0.01</td></tr>
<tr><td>0.1</td><td>0.08</td><td>9e-3</td><td>2e-6</td><td>6e-14±9e-15</td><td>0.1</td><td>0.95±0.01</td><td>0.99±0.01</td><td>0.99±0.01</td></tr>
<tr><td>0.01</td><td>1e-5</td><td>2e-8</td><td>4e-14</td><td>1e-14±4e-14</td><td>0.01</td><td>0.99±0.01</td><td>1.00±0.00</td><td>1.00±0.00</td></tr>
</table>

frames opponent shaping as a meta-learning problem requiring only first-order derivatives. M-FOS is consistent in that it does not make any assumption about the learning algorithm of the opponent. However, the work does not investigate consistency specifically.

3. Background

3.1. Differentiable games

The framework of differentiable games has become increasingly popular to model multi-agent learning. Whereas stochastic games are limited to parameters such as action-state probabilities, differentiable games generalize to any real-valued parameter vectors and differentiable loss functions (Balduzzi et al., 2018). We restrict our attention to two-player games, as is standard in much of the literature (Foerster et al., 2018a;b; Schäfer & Anandkumar, 2019).

Definition 3.1 (Differentiable games). In a two-player differentiable game, players $i = 1, 2$ control parameters $\theta_i \in \mathbb{R}^{d_i}$ to minimize twice continuously differentiable losses $L^i : \mathbb{R}^{d_1 + d_2} \to \mathbb{R}$. We adopt the convention to write $-i$ to denote the opponent of player i.

A fundamental challenge of the multi-loss setting is finding a good solution concept. Whereas in the single loss setting the typical solution concept are local minima, in multi-loss settings there are different sensible solution concepts. Most prominently, there are Nash Equilibria (Osborne & Rubinstein, 1994). However, Nash Equilibria include unstable saddle points that cannot be reasonably found via gradient-based learning algorithms (Letcher et al., 2019b). A more suitable concept are stable fixed points (SFPs), which could be considered a differentiable game analogon to local minima in single loss optimization. We will omit a formal definition here for brevity and point the reader to previous work on the topic (Letcher et al., 2019a).

3.2. LOLA

Consider a differentiable game with two players. A LOLA agent θ_1 uses its access to the opponent's parameters θ_2 to differentiate through a learning step of the opponent. That is,

agent 1 reformulates their loss to $\tilde{L}^1 = L^1\left(\theta_1, \theta_2 + \widehat{\Delta\theta_2}\right)$, where $\widehat{\Delta\theta_2}$ represents the assumed learning step of the opponent. In first-order LOLA we assume the opponent to be a naive learner: $\widehat{\Delta\theta_2} = -\alpha\nabla_2 L^2$. This assumption makes LOLA inconsistent when the opponent is any other type of learner. Here, ∇_2 denotes the gradient with respect to θ_2, and α represents the *look-ahead* rate, which is the *assumed* learning rate of the opponent. This rate may differ from the opponent's actual learning rate, but we will only consider equal learning rates and look-ahead rates across opponents for simplicity. In the original paper the loss was approximated using a Taylor expansion, $\tilde{L}^1 \approx L^1 + (\nabla_2 L^1)^\top \widehat{\Delta\theta_2}$. For agent 1, their first-order *Taylor* LOLA update is then

$$\Delta\theta_1 = -\alpha\left(\nabla_1 L^1 + \nabla_{12} L^1 \widehat{\Delta\theta_2} + \left(\nabla_1\widehat{\Delta\theta_2}\right)^\top \nabla_2 L^1\right).$$

Alternatively, in *exact* LOLA, the derivative is taken directly with respect to the reformulated loss, yielding the update

$$\Delta\theta_1 = -\alpha\nabla_1\left(L^1\left(\theta_1, \theta_2 + \widehat{\Delta\theta_2}\right)\right).$$

LOLA has had some empirical success, being one of the first general learning methods to discover tit-for-tat in the IPD. However, later work showed that LOLA does not preserve SFPs $\bar{\theta}$, e.g., the rightmost term in the equation for Taylor LOLA can be nonzero at $\bar{\theta}$. In fact, LOLA agents show "arrogant" behavior: they assume they can shape the learning of their *naive* opponents without having to adapt to the shaping of the opponent. Prior work hypothesized that this arrogant behavior is due to LOLA's inconsistent formulation and may be the cause for LOLA's failure to preserve SFPs (Letcher (2018), pp. 2, 26; Letcher et al. (2019b))

3.3. CGD

CGD (Schäfer & Anandkumar, 2019) proposes updates that are themselves Nash Equilibra of a local bilinear approximation of the game. It stands out by its robustness to different look-ahead rates and its ability to find SFPs. However, CGD does not find tit-for-tat on the IPD, instead converging to mutual defection (see Figure 3e). CGD's update rule is
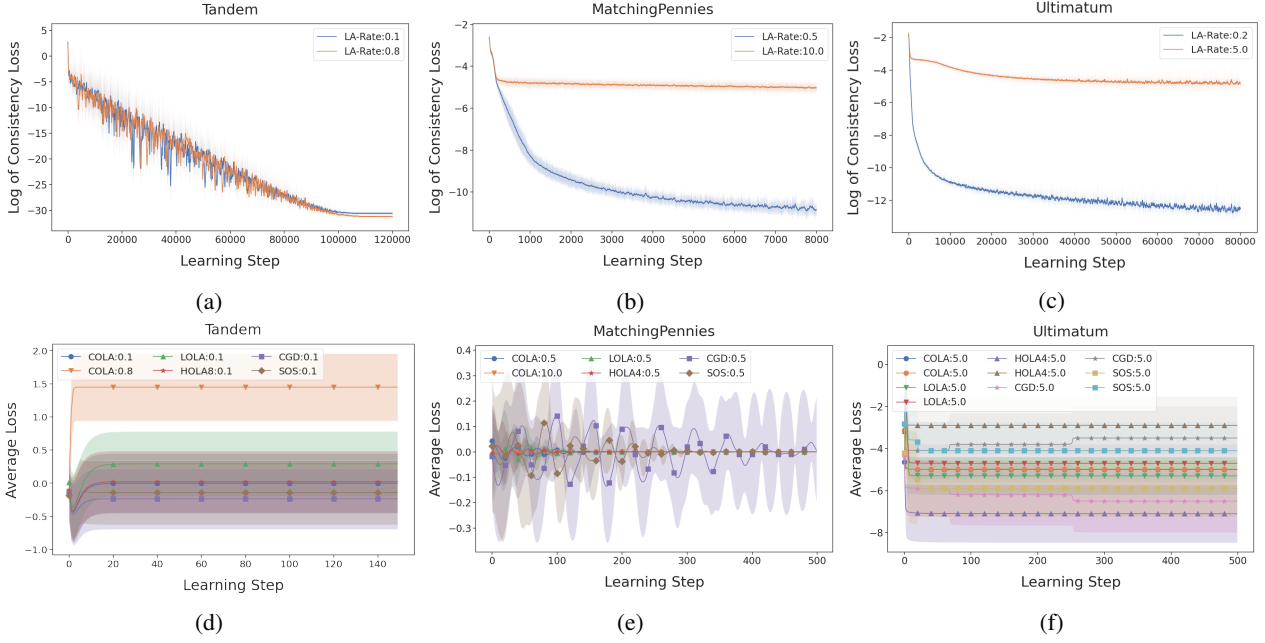
Figure 1. Subfigures (a), (b) and (c): Log of the consistency loss with standard error of 10 independent training runs over the training of the update functions for the Tandem, MP and Ultimatum games. Subfigures (d), (e) and (f): Learning outcomes for the respective games. E.g., "COLA:0.1" is COLA at a look-ahead rate of 0.1. Lines represent payoff means and shaded areas standard deviations over 10 runs.

given by

$$\begin{pmatrix} \Delta\theta_1 \\ \Delta\theta_2 \end{pmatrix} = -\alpha \begin{pmatrix} \mathrm{Id} & \alpha\nabla_{12}L^1 \\ \alpha\nabla_{21}L^2 & \mathrm{Id} \end{pmatrix}^{-1} \begin{pmatrix} \nabla_1 L^1 \\ \nabla_2 L^2 \end{pmatrix}.$$

One can recover different orders of CGD by approximating the inverse matrix via the series expansion $(\mathrm{Id} - A)^{-1} = \lim_{N\to\infty} \sum_{k=0}^{N} A^k$ for $\|A\| < 1$. For example, at N=1, we recover a version called Linearized CGD (LCGD), defined via $\Delta\theta_1 := -\alpha\nabla_1 L^1 + \alpha^2 \nabla_{12}L^1 \nabla_2 L^2$.

4. Method and theory

In this section, we formally define iLOLA and consistency under mutual opponent shaping and show that iLOLA is consistent, thus in principle addressing LOLA's inconsistency problem. We then clear up the relation between CGD and iLOLA, *correcting a false claim* in Schäfer & Anandkumar (2019). Lastly, we introduce COLA as an alternative to iLOLA and present some initial theoretical analysis, including the result that, contrary to prior belief, even consistent update functions do not recover SFPs.

4.1. Convergence and consistency of higher-order LOLA

The original formulation of LOLA is inconsistent when two LOLA agents learn together, because LOLA agents assume their opponent is a naive learner. To address this problem, we define and analyze iLOLA. In this section, we focus on

exact LOLA, but we provide a version of our analysis for Taylor LOLA in Appendix C. HOLAn is defined by the recursive relation

$$h_1^{n+1} := -\alpha\nabla_1\left(L^1(\theta_1, \theta_2 + h_2^n)\right)$$
$$h_2^{n+1} := -\alpha\nabla_2\left(L^2(\theta_1 + h_1^n, \theta_2)\right)$$

with $h_1^{-1} = h_2^{-1} = 0$, omitting arguments (θ^1, θ^2) for convenience. In particular, HOLA0 coincides with simultaneous gradient descent while HOLA1 coincides with LOLA.

Definition 4.1 (iLOLA). If HOLA$n = (h_1^n, h_2^n)$ converges pointwise as $n \to \infty$, define

$$\mathrm{iLOLA} := \lim_{n\to\infty} \begin{pmatrix} h_1^n \\ h_2^n \end{pmatrix} \text{ as the limiting update.}$$

We show in Appendix A that HOLA does not always converge, even in simple quadratic games. But, unlike LOLA, iLOLA satisfies a criterion of *consistency* whenever HOLA does converge (under some assumptions), formally defined as follows:

Definition 4.2 (Consistency). Any update functions $f_1 \colon \mathbb{R}^d \to \mathbb{R}^{d_1}$ and $f_2 \colon \mathbb{R}^d \to \mathbb{R}^{d_2}$ are *consistent* (under mutual opponent shaping with look-ahead rate α) if for all $\theta_1 \in \mathbb{R}^{d_1}, \theta_2 \in \mathbb{R}^{d_2}$, they satisfy

$$f_1(\theta_1, \theta_2) = -\alpha\nabla_1(L^1(\theta_1, \theta_2 + f_2(\theta_1, \theta_2))) \quad (1)$$
$$f_2(\theta_1, \theta_2) = -\alpha\nabla_2(L^2(\theta_1 + f_1(\theta_1, \theta_2), \theta_2)) \quad (2)$$

Table 2. On the MP game: Over different look-ahead rates we compare (a) the consistency losses and (b) the cosine similarity between COLA and LOLA, HOLA2, and HOLA4. The values represent the mean over 1,000 samples, uniformly sampled from the parameter space Θ. The error bars represent one standard deviation and capture the variance over 10 different COLA training runs.

(a)						(b)			
α	LOLA	HOLA2	HOLA4	COLA		α	LOLA	HOLA2	HOLA4
10	0.06	0.70	6.56	2e-3±3e-4		10	0.90±0.01	0.84±0.01	0.74±0.02
5	5e-3	0.03	0.15	5e-4±5e-5		5	0.98±0.01	0.97±0.01	0.92±0.01
1.0	9e-6	3e-8	4e-9	3e-6±1e-6		1.0	1.00±0.00	1.00±0.00	1.00±0.00
0.5	5e-7	3e-10	5e-12	3e-6±3e-6		0.5	1.00±0.00	1.00±0.00	1.00±0.00
0.01	1e-13	6e-17	5e-17	2e-6±2e-6		0.01	1.00±0.00	1.00±0.00	1.00±0.00

Proposition 4.3. *Let* $\text{HOLA}n = (h_1^n, h_2^n)$ *denote both players' exact n-th order LOLA updates. Assume that* $\lim_{n\to\infty} h_i^n(\theta) = h_i(\theta)$ *and* $\lim_{n\to\infty} \nabla_i h_{-i}^n(\theta) = \nabla_i h_{-i}(\theta)$ *exist for all* $\theta \in \mathbb{R}^d$ *and* $i \in \{1, 2\}$. *Then iLOLA is consistent under mutual opponent shaping.*

Proof. In Appendix B. □

4.2. CGD does not recover higher-order LOLA

Schäfer & Anandkumar (2019) claim that "LCGD [Linearized CGD] coincides with first order LOLA" (page 6), and moreover that the higher-order "series-expansion [of CGD] would recover higher-order LOLA" (page 4). If this were correct, it would imply that *full* CGD is equal to iLOLA and thus provides a convenient closed-form solution. We prove that this is false in general games:

Proposition 4.4. *CGD is inconsistent and does not coincide with iLOLA. In particular, Linearized CGD (LCGD) does not coincide with LOLA and the series-expansion of CGD does not recover HOLA (neither exact nor Taylor). Instead, LCGD coincides with LookAhead (Zhang & Lesser, 2010), an algorithm that lacks opponent shaping, and the series-expansion of CGD recovers higher-order LookAhead.*

Proof. In Appendix D. For the negative results, it suffices to construct a single counterexample: we show that LCGD and LOLA differ almost everywhere in the Tandem game (excluding a set of measure zero). We prove by contradiction that the series-expansion of CGD does not recover HOLA. If it did, CGD would equal iLOLA, and by Proposition 4.3, CGD would satisfy the consistency equations. However, this fails almost everywhere in the Tandem game, concluding the contradiction. □

4.3. COLA

iLOLA is consistent under mutual opponent shaping. However, HOLA does not always converge and, even when it does, it may be expensive to recursively compute HOLAn for sufficiently high n to achieve convergence. As an alternative, we propose COLA.

COLA learns consistent update functions and avoids infinite regress by directly solving the equations in Definition 4.2. We define the *consistency losses* for learned update functions f_1, f_2 parameterized by ϕ_1, ϕ_2, obtained for a given θ as the difference between RHS and LHS in Definition 4.2:

$$C_1(\phi_1, \phi_2, \theta_1, \theta_2) = \left\| f_1 + \alpha \nabla_1(L^1(\theta_1, \theta_2 + f_2)) \right\|$$
$$C_2(\phi_1, \phi_2, \theta_1, \theta_2) = \left\| f_2 + \alpha \nabla_2(L^2(\theta_1 + f_1, \theta_2)) \right\|.$$

If both losses are 0 for all θ, then the two update functions defined by ϕ_1, ϕ_2 are consistent. For this paper, we parameterise f_1, f_2 as neural networks with parameters ϕ_1, ϕ_2 respectively, and numerically minimize the sum of both losses over a region of interest.

The parameter region of interest Θ depends on the game being played. For games with probabilities as actions, we select an area that captures most of the probability space (e.g. we sample a pair of parameters $\theta_1, \theta_2 \sim [-7, 7]$, since $\sigma(7) \approx 1$ where σ is the sigmoid function).

We optimize the mean of the sum of consistency losses,

$$C(\phi_1, \phi_2) := \mathbb{E}_{(\theta_1, \theta_2) \sim \mathcal{U}(\Theta)} \big[C_1(\phi_1, \phi_2, \theta_1, \theta_2) + C_2(\phi_1, \phi_2, \theta_1, \theta_2) \big],$$

by sampling parameter pairs (θ_1, θ_2) uniformly from Θ and feeding them to the neural networks f_1, f_2, each outputting an agent's parameter update. The weights ϕ_1, ϕ_2 are then updated by taking a gradient step to minimize C. We train the update functions until the loss has converged and use the learned update functions to train a pair of agent policies in the given game.

4.4. Theoretical results for COLA

In this section, we provide some initial theoretical results for COLA's uniqueness and convergence behavior, using the Tandem game (Letcher et al., 2019b) and the Hamiltonian game (Balduzzi et al., 2018) as examples. These are simple polynomial games, with losses given in Section 5. Proofs for the following propositions can be found in Appendices E, F and G, respectively.
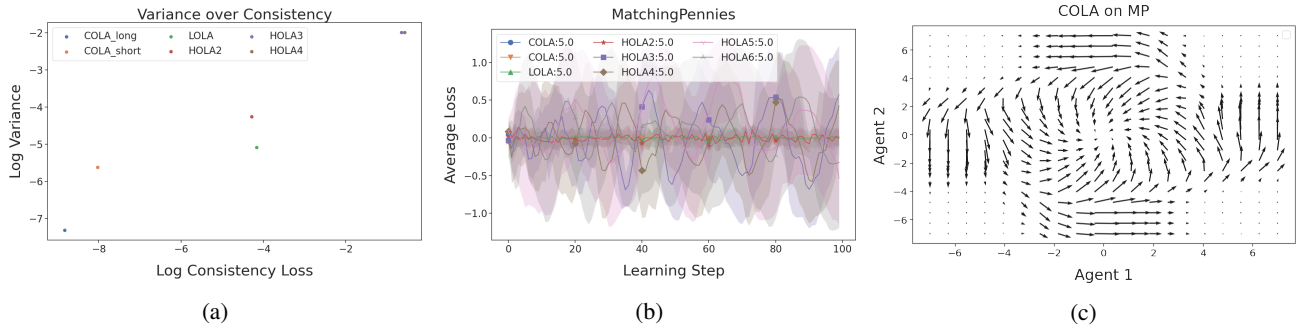
Figure 2. Training in MP at look-ahead rate of $\alpha = 10$. (a) Axes are on a log-scale. Shown is the mean variance and consistency over 10 different runs. Each run, COLA was retrained. COLA_long was trained for 80k steps and COLA_short for 800 steps. (b) LOLA and HOLA find non-convergent or even diverging solutions, while COLA's converge. (c) Gradient field learned by COLA on MP at a look-ahead rate of 10.

First, we show that solutions to the consistency equations are in general not unique, even when restricting to linear update functions in the Tandem game. Interestingly, empirically, COLA does seem to consistently converge to similar solutions regardless (see Table 7 in Appendix I.3).

Proposition 4.5. *Solutions to the consistency equations are not unique, even when restricted to linear solutions; more precisely, there exist several linear consistent solutions to the Tandem game.*

Second, we show that consistent solutions do not, in general, preserve SFPs, contradicting the hypothesis that LOLA's failure to preserve SFPs is due to its inconsistency (Letcher (2018), pp. 2, 26; Letcher (2018)). We experimentally confirm this result in Section 6.

Proposition 4.6. *Consistency does not imply preservation of SFPs: there is a consistent solution to the Tandem game with $\alpha = 1$ that fails to preserve **any** SFP. Moreover, for any $\alpha > 0$, there are **no** linear consistent solutions to the Tandem game that preserve more than one SFP.*

Third, we show that COLA can have more robust convergence behavior than LOLA and SOS:

Proposition 4.7. *For any non-zero initial parameters and any $\alpha > 1$, LOLA and SOS have divergent iterates in the Hamiltonian game. By contrast, any linear solution to the consistency equations converges to the origin for any initial parameters and **any** look-ahead rate $\alpha > 0$; moreover, the speed of convergence strictly increases with α.*

5. Experiments

We perform experiments on a set of games from the literature (Balduzzi et al., 2018; Letcher et al., 2019b) using LOLA, SOS and CGD as baselines. For details on the training procedure of COLA, we refer the reader to Appendix H.

First, we compare HOLA and COLA on polynomial general-sum games, including the Tandem game (Letcher et al., 2019b), where LOLA fails to converge to SFPs. Second, we investigate non-polynomial games, specifically the zero-sum Matching Pennies (MP) game, the general-sum Ultimatum game (Hutter, 2021) and the IPD (Axelrod & Hamilton, 1981; Harper et al., 2017).

Polynomial games. Losses in the **Tandem game** (Letcher et al., 2019b) are given by $L^1(x, y) = (x + y)^2 - 2x$ and $L^2(x, y) = (x+y)^2 - 2y$ for agent 1 and 2 respectively. The Tandem game was introduced to show that LOLA fails to preserve SFPs at $x + y = 1$ and instead converges to Pareto-dominated solutions (Letcher et al., 2019b). Additionally to the Tandem game, we investigate the algorithms on the **Hamiltonian game**, $L^1(x, y) = xy$ and $L^2(x, y) = -xy$; and the **Balduzzi game**, where $L^1(x, y) = \frac{1}{2}x^2 + 10xy$ and $L^2(x, y) = \frac{1}{2}y^2 - 10xy$ (Balduzzi et al., 2018).

Matching Pennies. The payoff matrix for MP (Lee & K, 1967) is shown in Appendix I.3 in Table 6. Each policy is parameterized with a single parameter, the log-odds of choosing heads $p_{\text{heads}} = \sigma(\theta_A)$. In this game, the unique Nash equilibrium is playing heads half the time.

Ultimatum game. The binary, single-shot Ultimatum game (Güth et al., 1982; Sanfey et al., 2003; Oosterbeek et al., 2004; Henrich et al., 2006) is set up as follows. Player 1 has access to $10. They can split the money fairly with player 2 ($5 for each player) or they can split it unfairly ($8 for player 1, $2 for player 2). Player 2 can either accept or reject the proposed split. If player 2 rejects, the reward is 0 for both players. If player 2 accepts, the reward follows the proposed split. Player 1's parameter is the log-odds of proposing a fair split $p_{\text{fair}} = \sigma(\theta_1)$. Player 2's parameter is the log-odds of accepting the unfair split (assuming that player 2 always accepts fair splits) $p_{\text{accept}} = \sigma(\theta_2)$. In terms
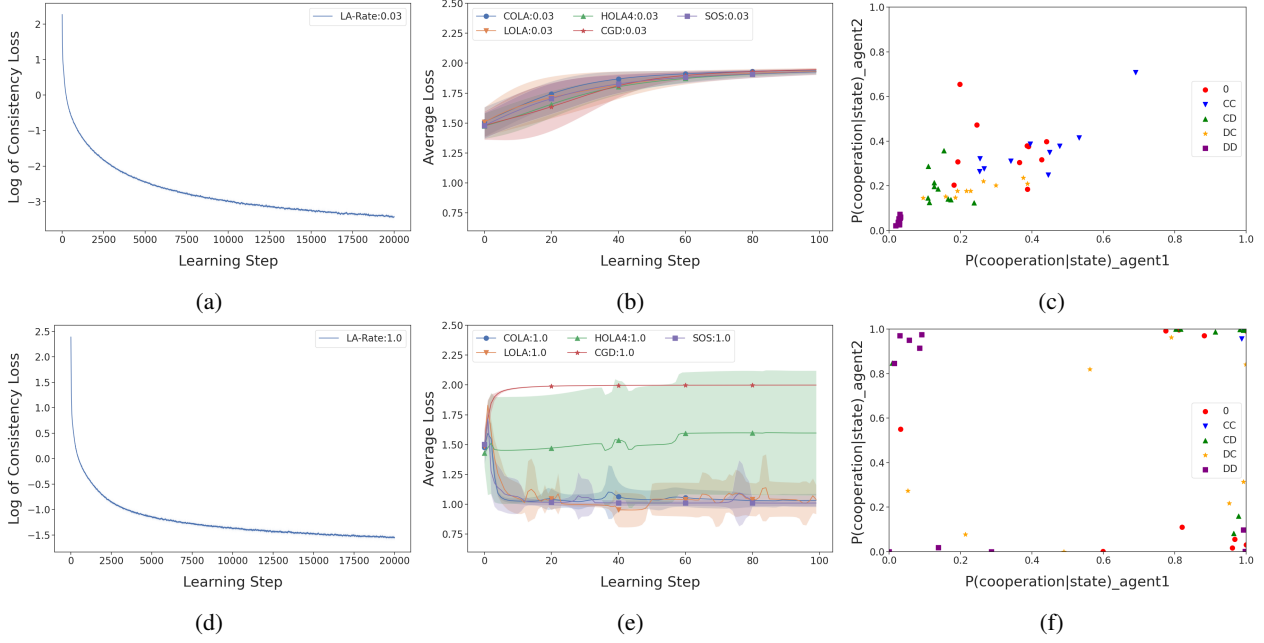
Figure 3. IPD results: Subfigure (a) / (d) show the consistency loss with shaded standard error over 10 independent training runs for look-ahead rates of 0.03 / 1.0, (b) / (e) the average loss and (c) / (f) the policy for the first player, both for the same pair of look-ahead rates. At low look-ahead HOLA defects and at high ones it diverges, also leading to high loss.

of losses, we have

$$L_1 = -\left(5p_{\text{fair}} + 8(1 - p_{\text{fair}})p_{\text{accept}}\right)$$
$$L_2 = -\left(5p_{\text{fair}} + 2(1 - p_{\text{fair}})p_{\text{accept}}\right).$$

IPD. We investigate the infinitely iterated Prisoner's Dilemma (IPD) (Axelrod & Hamilton, 1981; Harper et al., 2017) with discount factor $\gamma = 0.96$ and the usual payout function (see Appendix I.6). An agent i is defined through 5 parameters, the log-odds of cooperating in the first time step and across each of the four possible tuples of past actions of both players in the later steps.

6. Results

First, we report and compare the learning outcomes achieved by COLA and our baselines. We find that COLA update functions converge even under high look-ahead rates and learn socially desirable solutions. We also confirm our theoretical result (Proposition 4.4) that *CGD does not equal iLOLA*, contradicting Schäfer & Anandkumar (2019), and that COLA does not, in general, maintain SFPs (Proposition 4.6), contradicting the prior belief that this shortcoming is caused by inconsistency.

Second, we provide a more in-depth empirical analysis and comparison of the COLA and HOLA update functions, showing that COLA and HOLA tend to coincide when the latter converges, and that COLA is able to find consistent

solutions even when HOLA diverges. Moreover, while COLA's solutions are not unique in theory (Proposition 4.5), we empirically find that in our examples COLA tends to find similar solutions across different independent training runs. Additional results supporting the above findings are reported in Appendix I.

Learning Outcomes. In the Tandem game (Figure 1d), we see that COLA and HOLA8 converge to similar outcomes in the game, whereas CGD does not. This supports our theoretical result that CGD does not equal iLOLA (Proposition 4.4). We also see that COLA does not recover SFPs, thus experimentally confirming Proposition 4.6. In contrast to LOLA, HOLA and SOS, COLA finds a convergent solution even at a high look-ahead rate (see COLA:0.8 in Figure 1d and Figure 4b in Appendix I.1). CGD is the only other algorithm in the comparison that also shows robustness to high look-ahead rates in the Tandem game.

On the IPD, all algorithms find the defect-defect strategy on low look-ahead rates (Figure 3b). At high look-ahead rates, COLA finds a strategy qualitatively similar to tit-for-tat, as displayed in Figure 3f, though more noisy. However, COLA still achieves close to the optimal total loss, in contrast to CGD, which finds defect-defect even at a high look-ahead rate (see Figure 13 in Appendix I.6). The fact that, unlike HOLA and COLA, CGD finds defect-defect, further confirms that CGD does not equal iLOLA.

On MP at high look-ahead rates, SOS and LOLA mostly

Table 3. IPD: Over multiple look-ahead rates we compare (a) the consistency losses and (b) the cosine similarity between COLA and LOLA, HOLA2, and HOLA4. The values represent the mean over 250 samples, uniformly sampled from the parameter space Θ. The error bars represent one standard deviation and capture the variance over 10 different COLA training runs.

<table>
<tr><td colspan="5" align="center">(a)</td><td colspan="4" align="center">(b)</td></tr>
<tr><td>α</td><td>LOLA</td><td>HOLA2</td><td>HOLA4</td><td>COLA</td><td>α</td><td>LOLA</td><td>HOLA2</td><td>HOLA4</td></tr>
<tr><td>1.0</td><td>39.56</td><td>21.16</td><td>381.21</td><td>1.06±0.09</td><td>1.0</td><td>0.73±0.02</td><td>0.63±0.01</td><td>0.46±0.03</td></tr>
<tr><td>0.03</td><td>2e-3</td><td>5e-6</td><td>9e-8</td><td>0.16±0.02</td><td>0.03</td><td>0.97±0.01</td><td>0.97±0.01</td><td>0.97±0.01</td></tr>
</table>

don't converge, whereas COLA converges *even faster* with a high look-ahead rate (see Figure 2a), confirming Proposition 4.7 experimentally (also see Figure 6b and 7b in Appendix I.2). To further investigate the influence of consistency on learning behavior, we plot the consistency of an update function against the variance of the losses across learning steps achieved by that function, for different orders of HOLA and for COLA (Figure 2b). At a high look-ahead rate in Matching Pennies, we find that more consistent update functions tend to lead to lower variance across training, demonstrating a potential benefit of increased consistency at least at high look-ahead rates.

For the Ultimatum game, we find that COLA is the only method that finds the fair solution consistently at a high look-ahead rate, whereas SOS, LOLA, and CGD do not (Figure 1f). At low look-ahead rates, all algorithms find the unfair solution (see Figure 10b in Appendix I.4). This demonstrates an advantage of COLA over our baselines and shows that higher look-ahead rates can lead to better learning outcomes.

Lastly, we introduce the Chicken game in Appendix I.5. Both Taylor LOLA and SOS crash, whereas COLA, HOLA, CGD, and exact LOLA swerve at high look-ahead rates (Figure 12d). Crashing in Chicken results in a catastrophic payout for both agents, whereas swerving results in a jointly preferable outcome.[1]

Update functions. Turning to our analysis of COLA and HOLA update functions, we first investigate how increasing the order of HOLA affects the consistency of its updates. As shown in Table 1a, 2a and 3a, HOLA's updates become more consistent with increasing order, but only below a certain, game-specific look-ahead rate threshold. Above that threshold, HOLA's updates become less consistent with increasing order.

Second, we compare the consistency losses of COLA and HOLA. In the aforementioned tables, we observe that COLA achieves low consistency losses on most games. Below the threshold, COLA finds similarly low consistency losses

---

[1]Interestingly, in contrast to Taylor LOLA, exact LOLA swerves. The Chicken game is the only game where we found a difference in learning behavior between exact LOLA and Taylor LOLA.

as HOLA, though there HOLA's are lower in the non-polynomial games. Above the threshold, COLA finds consistent updates, even when HOLA does not. A visualization of the update function learned by COLA at a high look-ahead rate on the MP is given in Figure 2c.

For the IPD, COLA's consistency losses are high compared to other games, but much lower than HOLA's consistency losses at high look-ahead rates. We leave it to future work to find methods that obtain more consistent solutions.

Third, we are interested whether COLA and HOLA find similar solutions. We calculate the cosine similarity between the respective update functions over Θ. As we show in Table 1b, 2b and 3b, COLA and HOLA find very similar solutions when HOLA's updates converge, i.e., when the look-ahead rate is below the threshold. Above the threshold, COLA's and HOLA's updates unsurprisingly become less similar with increasing order, as HOLA's updates diverge with increasing order.

Lastly, we investigate Proposition 4.5 empirically and find that COLA finds similar solutions in Tandem and MP over 5 training runs (see Table 7 in Appendix I.3). Moreover, the small standard deviations in Table 3b indicate that COLA also finds similar solutions over different runs in the IPD.

7. Conclusion and Future Work

In this paper, we corrected a claim made in prior work (Schäfer & Anandkumar, 2019), clearing up the relation between the CGD and LOLA algorithms. We also showed that iLOLA solves part of the consistency problem of LOLA. We introduced COLA, which finds consistent solutions without requiring many recursive computations like iLOLA. It was believed that inconsistency leads to arrogant behaviour and lack of preservation of SFPs. We showed that even with consistency, opponent shaping behaves *arrogantly*, pointing towards a fundamental open problem for the method.

In a set of games, we found that COLA tends to find prosocial solutions. Although COLA's solutions are not unique in theory, empirically, COLA tends to find similar solutions in different runs. It coincides with iLOLA when HOLA converges and finds consistent update functions even when HOLA fails to converge with increasing order. Moreover, we showed empirically (and in one case theoretically) that

COLA update functions converge under a wider range of look-ahead rates than HOLA and LOLA update functions.

This work raises many questions for future work, such as the existence of solutions to the COLA equations in general games and general properties of convergence and learning outcomes. Moreover, additional work is needed to scale COLA to large settings such as GANs or Deep RL, or settings with more than two players. Another interesting axis is addressing further inconsistent aspects of LOLA as identified in Letcher et al. (2019b).

Acknowledgements

References

Albrecht, S. V. and Stone, P. Reasoning about hypothetical agent behaviours and their parameters. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 547–555, 2017.

Axelrod, R. and Hamilton, W. D. The evolution of cooperation. *Science*, 211(4489):1390–1396, 1981.

Azizian, W., Mitliagkas, I., Lacoste-Julien, S., and Gidel, G. A tight and unified analysis of gradient-based methods for a whole spectrum of differentiable games. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2863–2873, 2020.

Balduzzi, D., Racanière, S., Martens, J., Foerster, J. N., Tuyls, K., and Graepel, T. The mechanics of n-player differentiable games. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 363–372, 2018.

Barto, A. G. and Mahadevan, S. Recent advances in hierarchical reinforcement learning. *Discret. Event Dyn. Syst.*, 13(1-2):41–77, 2003.

Dafoe, A., Hughes, E., Bachrach, Y., Collins, T., McKee, K. R., Leibo, J. Z., Larson, K., and Graepel, T. Open problems in cooperative ai. In *Cooperative AI workshop*, 2021.

Foerster, J., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., and Mordatch, I. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 122–130, 2018a.

Foerster, J. N., Farquhar, G., Al-Shedivat, M., Rocktäschel, T., Xing, E. P., and Whiteson, S. Dice: The infinitely differentiable monte carlo estimator. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1524–1533, 2018b.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, pp. 2672–2680, 2014.

Güth, W., Schmittberger, R., and Schwarze, B. An experimental analysis of ultimatum bargaining. *Journal of Economic Behavior & Organization*, 3(4):367–388, 1982.

Harper, M., Knight, V., Jones, M., Koutsovoulos, G., Glynatsi, N. E., and Campbell, O. Reinforcement learning produces dominant strategies for the iterated prisoner's dilemma. *PLOS ONE*, 12(12):e0188046, 2017.

He, H. and Boyd-Graber, J. L. Opponent modeling in deep reinforcement learning. In *Proceedings of the 33nd International Conference on Machine Learning*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1804–1813, 2016.

Henrich, J., Boyd, R., Bowles, S., Camerer, C., Fehr, E., and Gintis, H. Foundations of Human Sociality: Economic Experiments and Ethnographic Evidence From Fifteen Small-Scale Societies. In *American Anthropologist*, volume 108. 2006.

Hutter, A. Learning in two-player games between transparent opponents. arXiv preprint arXiv:2012.02671, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, 2015.

Korpelevich, G. M. The extragradient method for finding saddle points and other problems. volume 13, pp. 35–49, 1977.

Lee, K. and K, L. *The Application of Decision Theory and Dynamic Programming to Adaptive Control Systems*. Thesis, 1967.

Letcher, A. Stability and exploitation in differentiable games. Master's thesis, University of Oxford, 2018.

Letcher, A., Balduzzi, D., Racanière, S., Martens, J., Foerster, J. N., Tuyls, K., and Graepel, T. Differentiable game mechanics. *J. Mach. Learn. Res.*, 20:84:1–84:40, 2019a.

Letcher, A., Foerster, J. N., Balduzzi, D., Rocktäschel, T., and Whiteson, S. Stable opponent shaping in differentiable games. In *7th International Conference on Learning Representations*, 2019b.

Lu, C., Willi, T., Schroeder de Witt, C., and Foerster, J. Model-free opponent shaping. *arXiv preprint arXiv:2205.01447*, 2022.

Mazumdar, E. V., Jordan, M. I., and Sastry, S. S. On finding local nash equilibria (and only local nash equilibria) in zero-sum games. arXiv preprint arXiv:1901.00838, 2019.

Mealing, R. and Shapiro, J. L. Opponent modeling by expectation-maximization and sequence prediction in simplified poker. *IEEE Trans. Comput. Intell. AI Games*, 9(1):11–24, 2017.

Mescheder, L. M., Nowozin, S., and Geiger, A. The numerics of gans. In *Advances in Neural Information Processing Systems*, volume 30, pp. 1825–1835, 2017.

Oosterbeek, H., Sloof, R., and van de Kuilen, G. Cultural Differences in Ultimatum Game Experiments: Evidence from a Meta-Analysis. *Experimental Economics*, 7(2): 171–188, 2004.

Oroojlooyjadid, A. and Hajinezhad, D. A review of cooperative multi-agent deep reinforcement learning. 2019.

Osborne, M. J. and Rubinstein, A. *A Course in Game Theory*. The MIT Press, Cambridge, MA, 1994.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pp. 8024–8035. 2019.

Racanière, S., Weber, T., Reichert, D. P., Buesing, L., Guez, A., Rezende, D. J., Badia, A. P., Vinyals, O., Heess, N., Li, Y., Pascanu, R., Battaglia, P. W., Hassabis, D., Silver, D., and Wierstra, D. Imagination-augmented agents for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5690–5701, 2017.

Sanfey, A. G., Rilling, J. K., Aronson, J. A., Nystrom, L. E., and Cohen, J. D. The Neural Basis of Economic Decision-Making in the Ultimatum Game. *Science*, 300(5626): 1755–1758, 2003.

Schäfer, F. and Anandkumar, A. Competitive gradient descent. In *Advances in Neural Information Processing Systems*, volume 32, pp. 7623–7633, 2019.

Schmidhuber, J. A possibility for implementing curiosity and boredom in model-building neural controllers. In Meyer, J. A. and Wilson, S. W. (eds.), *Proc. of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pp. 222–227. MIT Press/Bradford Books, 1991.

Schäfer, F., Anandkumar, A., and Owhadi, H. Competitive mirror descent. arXiv preprint arXiv:2006.10179, 2020.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T. P., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., and Hassabis, D. Mastering the game of go without human knowledge. *Nat.*, 550(7676):354–359, 2017.

Synnaeve, G. and Bessière, P. A bayesian model for opening prediction in RTS games with application to starcraft. In *IEEE Conference on Computational Intelligence and Games*, pp. 281–288, 2011.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., Vezhnevets, A. S., Leblond, R., Pohlen, T., Dalibard, V., Budden, D., Sulsky, Y., Molloy, J., Paine, T. L., Gülçehre, Ç., Wang, Z., Pfaff, T., Wu, Y., Ring, R., Yogatama, D., Wünsch, D., McKinney, K., Smith, O., Schaul, T., Lillicrap, T. P., Kavukcuoglu, K., Hassabis, D., Apps, C., and Silver, D. Grandmaster level in starcraft II using multi-agent reinforcement learning. *Nat.*, 575(7782):350–354, 2019.

Weber, B. G. and Mateas, M. A data mining approach to strategy prediction. In *Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Games*, pp. 140–147, 2009.

Wen, Y., Yang, Y., Luo, R., Wang, J., and Pan, W. Probabilistic recursive reasoning for multi-agent reinforcement learning. In *7th International Conference on Learning Representations*, 2019.

Zhang, C. and Lesser, V. R. Multi-agent learning with policy prediction. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

A. Nonconvergence of HOLA in the Tandem Game

In the following, we show that for the choice of look-ahead rate $\alpha = 1$, HOLA does not converge in the Tandem game. This shows that given a large enough look-ahead rate, even in a simple quadratic game, HOLA need not converge.

Proposition A.1. *Let L^1, L^2 be the two players' loss functions in the Tandem game as defined in Section 5:*

$$L^1(x, y) = (x + y)^2 - 2x \quad and \quad L^2(x, y) = (x + y)^2 - 2y, \tag{3}$$

and let h_i^n denote the n-th order exact LOLA update for player i (where $n = 0$ denotes naive learning). Consider the look-ahead rate $\alpha := 1$. Then the functions $(h_i^n)_{n \in \mathbb{N}}$ for $i = 1, 2$ do not converge pointwise.

Proof. We will prove the auxiliary statement that

$$h_i^n(x, y) = 2^{n+2} - 2(1 + x + y)$$

for $i = 1, 2$. It then follows trivially that the h_i^n cannot converge.

The auxiliary result can be proven by induction. The base case $n = 0$ follows from

$$\nabla_i L^i(x, y) = 2 - 2(x + y) = 2^2 - 2(1 + x + y)$$

for $i = 1, 2$. Next, for the inductive step, we have to show that

$$h_1^n(x, y) = -\nabla_1(L^1(x, y + h_2^{n-1}(x, y))) \tag{4}$$
$$h_2^n(x, y) = -\nabla_2(L^2(x + h_1^{n-1}(x, y), y)) \tag{5}$$

for any $n > 0$. Substituting the inductive hypothesis in the second step, we have

$$-\nabla_1(L^1(x, y + h_2^{n-1}(x, y))) \tag{6}$$
$$= -\nabla_1(L^1(x, y + 2^{n+1} - 2(1 + x + y))) \tag{7}$$
$$= -\nabla_1\left((x + y + 2^{n+1} - 2 - 2x - 2y)^2 - 2x\right) \tag{8}$$
$$= -\nabla_1\left((-x - y + 2^{n+1} - 2)^2 - 2x\right) \tag{9}$$
$$= 2(-x - y + 2^{n+1} - 2) + 2 \tag{10}$$
$$= 2^{n+2} - 2(1 + x + y) \tag{11}$$
$$= h_1^n(x, y). \tag{12}$$

The derivation for $h_2^n(x, y)$ is exactly analogous. This shows the inductive step and thus finishes the proof.

$\square$

B. Proof of Proposition 4.3

To begin, recall that some differentiable game with continuously differentiable loss functions L^1, L^2 is given, and that $h^n = (h_1^n, h_2^n)$ denotes the n-th order exact LOLA update function. We assume that the iLOLA update function h exists, defined via

$$h_i(\theta) := \lim_{n \to \infty} h_i^n(\theta),$$

for all $\theta \in \mathbb{R}^d$.

To prove Proposition 4.3, we need to show that h_1, h_2 are consistent, i.e., satisfy Definition 4.2, under the assumption that

$$\lim_{n \to \infty} \nabla_i h_{-i}^n(\theta) = \nabla_i h_{-i}(\theta)$$

for $i = 1, 2$ and any θ.

To that end, define the *(exact) LOLA operator* Ψ as the function mapping a pair of update functions $f := (f_1, f_2)$ to the RHS of Equations 1 and 2,

$$\Psi_1(f)(\theta) := -\alpha \nabla_1(L^1(\theta_1, \theta_2 + f_2(\theta_1, \theta_2))) \tag{13}$$

$$\Psi_2(f)(\theta) := -\alpha \nabla_2(L^2(\theta_1 + f_1(\theta_1, \theta_2), \theta_2)) \tag{14}$$

for any θ. Note that then we have $h_i^{n+1} = \Psi_i(h^n)$, i.e., Ψ maps n-th order LOLA to $n+1$-order LOLA.

In the following, we show that iLOLA is a fixed point of the LOLA operator, i.e., $\Psi(h) = h$. It follows from the definition of Ψ that then h is consistent. We denote by $\|\cdot\|$ the Euclidean norm or the induced operator norm for matrices. We focus on showing $\Psi_1(h) = h_1$. The case $i = 2$ is exactly analogous.

For arbitrary θ and n, define $\hat{\theta}_2 := \theta_2 + h_2(\theta)$ and $\hat{\theta}_2^n := \theta_2 + h_n^2(\theta)$ as the updated parameter of player 2. First, it is helpful to show that $\Psi_1(h^n)(\theta)$ converges to $\Psi_1(h)(\theta)$:

$$0 \leq \|\Psi_1(h)(\theta) - \Psi_1(h^n)(\theta)\| \tag{15}$$

$$= \alpha \|\nabla_1(L^1(\theta_1, \theta_2 + h_2(\theta))) - \nabla_1(L^1(\theta_1, \theta_2 + h_2^n(\theta)))\| \tag{16}$$

$$= \alpha \|(\nabla_1 h_2(\theta))^\top \nabla_2 L^1(\theta_1, \hat{\theta}_2) - (\nabla_1 h_2^n(\theta))^\top \nabla_2 L^1(\theta_1, \hat{\theta}_2^n) + \nabla_1 L^1(\theta_1, \hat{\theta}_2) - \nabla_1 L^1(\theta_1, \hat{\theta}_2^n)\| \tag{17}$$

$$\leq \alpha \|(\nabla_1 h_2(\theta))^\top \nabla_2 L^1(\theta_1, \hat{\theta}_2) - (\nabla_1 h_2^n(\theta))^\top \nabla_2 L^1(\theta_1, \hat{\theta}_2^n)\| + \alpha \|\nabla_1 L^1(\theta_1, \hat{\theta}_2) - \nabla_1 L^1(\theta_1, \hat{\theta}_2^n)\| \tag{18}$$

$$= \alpha \|(\nabla_1 h_2(\theta))^\top (\nabla_2 L^1(\theta_1, \hat{\theta}_2) - \nabla_2 L^1(\theta_1, \hat{\theta}_2^n)) \tag{19}$$
$$+ (\nabla_1 h_2(\theta) - \nabla_1 h_2^n(\theta))^\top \nabla_2 L^1(\theta_1, \hat{\theta}_2^n)\| + \alpha \|\nabla_1 L^1(\theta_1, \hat{\theta}_2) - \nabla_1 L^1(\theta_1, \hat{\theta}_2^n)\|$$

$$\leq \alpha \|(\nabla_1 h_2(\theta))^\top\| \|\nabla_2 L^1(\theta_1, \hat{\theta}_2) - \nabla_2 L^1(\theta_1, \hat{\theta}_2^n)\|$$
$$+ \alpha \|(\nabla_1 h_2(\theta) - \nabla_1 h_2^n(\theta))^\top\| \|\nabla_2 L^1(\theta_1, \hat{\theta}_2^n)\| + \alpha \|\nabla_1 L^1(\theta_1, \hat{\theta}_2) - \nabla_1 L^1(\theta_1, \hat{\theta}_2^n)\| \tag{20}$$

$$\overset{n\to\infty}{\longrightarrow} 0. \tag{21}$$

In the last step, we used the following two facts. First, since $\nabla_i L^1(\theta)$ is assumed to be continuous in θ_2, and $\lim_{n\to\infty} \hat{\theta}_2^n = \theta_2 + \lim_{n\to\infty} h_2^n(\theta) = \theta_2 + h_2(\theta) = \hat{\theta}_2$ by assumption, it follows that $\lim_{n\to\infty} \nabla_2 L^1(\theta_1, \hat{\theta}_2^n) = \nabla_2 L^1(\theta_1, \hat{\theta}_2)$ and $\lim_{n\to\infty} \nabla_1 L^1(\theta_1, \hat{\theta}_2^n) = \nabla_1 L^1(\theta_1, \hat{\theta}_2)$. Second, by assumption, $\lim_{n\to\infty} \nabla h_2^n(\theta) = \nabla h_2(\theta)$. In particular, $\|\nabla_2 L^1(\theta_1, \hat{\theta}_2^n)\|$ must be bounded, and thus the three terms in (20) must all converge to 0 as $n \to \infty$. It follows by the sandwich theorem that $\lim_{n\to\infty} \Psi_1(h^n)(\theta) = \Psi_1(h)(\theta)$.

Now we can directly prove that $\Psi_1(h)(\theta) = h_1(\theta)$. It is

$$0 \leq \|\Psi_1(h)(\theta) - h_1(\theta)\| \tag{22}$$

$$= \|\Psi_1(h)(\theta) - \Psi_1(h^n)(\theta) + \Psi_1(h^n)(\theta) - h_1^n(\theta) + h_1^n(\theta) - h_1(\theta)\| \tag{23}$$

$$\leq \|\Psi_1(h)(\theta) - \Psi_1(h^n)(\theta)\| + \|\Psi_1(h^n)(\theta) - h_1^n(\theta)\| + \|h_1^n(\theta) - h_1(\theta)\| \tag{24}$$

$$= \|\Psi_1(h)(\theta) - \Psi_1(h^n)(\theta)\| + \|h_1^{n+1}(\theta) - h_1^n(\theta)\| + \|h_1^n(\theta) - h_1(\theta)\| \tag{25}$$

$$\overset{n\to\infty}{\longrightarrow} 0, \tag{26}$$

where in the last step we have used the above result, as well as the assumption that $h_1^n(\theta)$ converges pointwise, and thus must also be a Cauchy sequence, so the last and the middle term both converge to zero as well.

It follows by the sandwich theorem that $\Psi_1(h)(\theta) = h_1(\theta)$. Since θ was arbitrary, this concludes the proof. $\square$

C. Infinite-order Taylor LOLA

In this Section, we repeat the analysis of iLOLA from Section 4.1 for infinite-order Taylor LOLA (Taylor iLOLA). I.e., we define Taylor consistency, and show that Taylor iLOLA satisfies this consistency equation under certain assumptions. This result will be needed for our proof of Proposition 4.4.

To begin, assume that some differentiable game with continuously differentiable loss functions L^1, L^2 is given. Define the Taylor LOLA operator Φ that maps pairs of update functions (f_1, f_2) to the associated Taylor LOLA update

$$\Phi_i(f) := -\alpha \nabla_i(L^i + (\nabla_{-i} L^i)^\top f_{-i}) \tag{27}$$

for $i = 1, 2$.

We then have the following definition.

Definition C.1 (Taylor consistency)**.** Two update functions f_1, f_2 are called Taylor consistent if for any $i = 1, 2$, we have

$$\Phi(f_1, f_2) = (f_1, f_2).$$

Next, let h_i^n denote i's n-th order Taylor LOLA update. I.e., $h_i^n := \Phi_i(h^{n-1})$ for $n \geq 0$, where we let $h_i^{-1} := 0$. Then we define

Definition C.2 (Taylor iLOLA)**.** If (h_1^n, h_2^n) converges pointwise as $n \to \infty$, define Taylor iLOLA as the limiting update

$$h := \lim_{n \to \infty} \begin{pmatrix} h_1^n \\ h_2^n. \end{pmatrix}$$

Finally, we provide a proof that Taylor iLOLA is Taylor consistent; i.e., we give a Taylor version of Proposition 4.3.

Proposition C.3. *Let h_i^n denote player i's n-th order Taylor LOLA update. Assume that $\lim_{n \to \infty} h_i^n(\theta) = h_i(\theta)$ and $\lim_{n \to \infty} \nabla_i h_{-i}^n(\theta) = \nabla_i h_{-i}(\theta)$ for all θ and $i \in \{1, 2\}$. Then Taylor iLOLA is Taylor consistent.*

Proof. The proof is exactly analogous to that of Proposition 4.3, but easier. We show $\Phi(h) = h$. It follows from the definition of Φ in Equation 27 that then h is Taylor consistent. We focus on showing $\Phi_1(h) = h_1$, and the case $i = 2$ is exactly analogous.

First, we show that $\Phi_1(h^n)(\theta)$ converges to $\Phi_1(h)(\theta)$ for all θ. Letting n be arbitrary and omitting θ in the following for clarity, it is

$$0 \leq \|\Phi_1(h) - \Phi_1(h^n)\| \tag{28}$$

$$= \| - \alpha \nabla_1(L^1 + (\nabla_2 L^1)^\top h_2) + \alpha \nabla_1(L^1 + (\nabla_2 L^1)^\top h_2^n)\| \tag{29}$$

$$= \alpha \| - \nabla_{12} L^1 h_2 - (\nabla_2 L^1)^\top (\nabla_1 h_2)^\top + \nabla_{12} L^1 h_2^n + (\nabla_2 L^1)^\top (\nabla_1 h_2^n)^\top\| \tag{30}$$

$$\leq \alpha \|\nabla_{12} L^1 (h_2^n - h_2)\| + \alpha \|(\nabla_2 L^1)^\top (\nabla_1 h_2^n - \nabla_1 h_2)^\top\| \tag{31}$$

$$\leq \alpha \|\nabla_{12} L^1\| \|h_2^n - h_2\| + \alpha \|\nabla_2 L^1\| \|\nabla_1 h_2^n - \nabla_1 h_2\| \tag{32}$$

$$\stackrel{n \to \infty}{\longrightarrow} 0. \tag{33}$$

In the last step, we used the assumptions that $\lim_{n \to \infty} h_2^n = h_2$ and $\lim_{n \to \infty} \nabla h_2^n = \nabla h_2$. It follows by the sandwich theorem that $\lim_{n \to \infty} \Phi_1(h^n)(\theta) = \Phi_1(h)(\theta)$.

It follows from the above that $\Phi_1(h)(\theta) = h_1(\theta)$, using exactly the same argument as in Equations 22-26 with Φ instead of Ψ. Since θ was arbitrary, this concludes the proof.

$\square$

D. Proof of Proposition 4.4

We begin by proving that LCGD does not coincide with Taylor LOLA and CGD neither coincides with exact nor Taylor iLOLA. It is sufficient to manifest a single counter-example: we consider the Tandem game given by $L^1 = (x + y)^2 - 2x$ and $L^2 = (x + y)^2 - 2y$ (using x, y instead of θ_1, θ_2 for simplicity). Throughout this proof we use the notation introduced by Balduzzi et al. (2018) and Letcher et al. (2019b) including the *simultaneous gradient*, the *off-diagonal Hessian* and the *shaping term* of the game as

$$\xi = \begin{pmatrix} \nabla_1 L^1 \\ \nabla_2 L^2 \end{pmatrix} \quad \text{and} \quad H_o = \begin{pmatrix} 0 & \nabla_{12} L^2 \\ \nabla_{21} L^2 & 0 \end{pmatrix} \quad \text{and} \quad \chi = \text{diag}\left(H_o^T \nabla L\right)$$

respectively. Note that in two-player games, Taylor LOLA's shaping term reduces to

$$\chi = \begin{pmatrix} \nabla_{12} L^2 \nabla_2 L^1 \\ \nabla_{21} L^1 \nabla_1 L^2 \end{pmatrix}.$$

LCGD $\neq$ LOLA. Following Schäfer & Anandkumar (2019), LCGD is given by

$$\text{LCGD} = -\alpha \begin{pmatrix} \nabla_x f - \alpha D^2_{xy} f \nabla_y g \\ \nabla_y g - \alpha D^2_{yx} g \nabla_x f \end{pmatrix} = -\alpha \begin{pmatrix} I & -\alpha D^2_{xy} f \\ -\alpha D^2_{yx} g & I \end{pmatrix} \begin{pmatrix} \nabla_x f \\ \nabla_y g \end{pmatrix} = -\alpha (I - \alpha H_o) \xi$$

while Taylor LOLA is given (Letcher et al., 2019b) by

$$\text{LOLA} = -\alpha (I - \alpha H_o) \xi + \alpha^2 \chi.$$

Any game with $\chi \neq 0$ will yield a difference between LCGD and LOLA; in particular,

$$\chi = 4(x + y) \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

in the Tandem game implies that LCGD $\neq$ LOLA whenever parameters lie outside the measure-zero set $\{x + y = 0\} \subset \mathbb{R}^2$.

CGD does not recover HOLA. Since CGD is obtained through a bilinear approximation (Taylor expansion) of the loss functions, one would expect that the authors' claim of recovering HOLA is with regards to Taylor (not exact) HOLA. For completeness, and to avoid any doubts for the reader, we prove that CGD neither corresponds to exact nor Taylor HOLA.

Following Schäfer & Anandkumar (2019), the series-expansion of CGD is given by

$$\text{CGD}n = -\alpha \sum_{i=0}^{n} \begin{pmatrix} 0 & -\alpha D^2_{xy} f \\ -\alpha D^2_{yx} g & 0 \end{pmatrix}^i \begin{pmatrix} D_x f \\ D_y g \end{pmatrix} = -\alpha \sum_{i=0}^{n} (-\alpha H_o)^i \xi$$

and converges to CGD whenever $\alpha < 1/\|H_o\|$ (where $\|\cdot\|$ denotes the operator norm induced by the Euclidean norm on the space). Assume for contradiction that the series-expansion of CGD recovers HOLA, i.e. that CGDn = HOLAn for all n. In particular, we must have

$$\text{CGD} = \lim_{n \to \infty} \text{CGD}n = \lim_{n \to \infty} \text{HOLA}n = \text{iLOLA}$$

whenever $\alpha < 1/\|H_o\|$. In the tandem game, we have

$$H_o = 2 \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

with $\|H_o\| = 2$, so CGD = iLOLA whenever $\alpha < 1/2$. Moreover, H_o being constant implies that

$$\nabla \text{HOLA}n = \nabla \text{CGD}n = -\alpha \sum_{i=0}^{n} (-\alpha H_o)^i \nabla \xi,$$

so gradients of HOLA also converge pointwise for all $\alpha < 1/2$. In particular, CGD = iLOLA must satisfy the (exact or Taylor) consistency equations by Proposition 4.3 or Proposition C.3. However, the update for CGD is given by

$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} = -\alpha (I + \alpha H_o)^{-1} \xi = -2\alpha(x + y - 1) \begin{pmatrix} 1 & 2\alpha \\ 2\alpha & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{-2\alpha(x + y - 1)}{1 + 2\alpha} \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

For the exact case, the RHS of the first consistency equation is

$$\begin{aligned}
-\alpha \nabla_x \left((x + y + f_2)^2 - 2x \right) &= -2\alpha \left((1 + \nabla_x f_2)(x + y + f_2) - 1 \right) \\
&= \frac{-2\alpha}{1 + 2\alpha} \left(x + y + \frac{-2\alpha(x + y - 1)}{1 + 2\alpha} - 1 - 2\alpha \right) \\
&= f_1 + \frac{4\alpha^2(x + y + 2\alpha)}{(1 + 2\alpha)^2}
\end{aligned}$$

which does not coincide with the LHS of the consistency equation ($= f_1$) whenever parameters lie outside the measure-zero set $\{x + y + 2\alpha = 0\} \subset \mathbb{R}^2$. Similarly for Taylor iLOLA, the RHS of the first consistency equation is

$$-\alpha \nabla_x \left((x+y)^2 - 2x + 2(x+y)f_2\right) = -2\alpha \left(x + y - 1 + \frac{-2\alpha(x+y-1)}{1+2\alpha} + \frac{-2\alpha(x+y)}{1+2\alpha}\right)$$
$$= f_1 + \frac{4\alpha^2(x+y)}{(1+2\alpha)^2}$$

which does not coincide with the LHS of the consistency equation ($= f_1$) whenever parameters lie outside the measure-zero set $\{x + y = 0\} \subset \mathbb{R}^2$. This is a contradiction to consistency; we are done.

LCGD = LookAhead. We have already shown that LCGD is given by $-\alpha(I - \alpha H_o)\xi$ in the proof that LCGD $\neq$ LOLA. This coincides exactly with LookAhead following Letcher et al. (2019b).

CGD recovers higher-order LookAhead. The series expansion of CGD is given by

$$\text{CGD}n = -\alpha \sum_{i=0}^{n} (-\alpha H_o)^i \xi \, .$$

Also, higher-order (Taylor) LookAhead is defined recursively by expanding

$$f_1^{n+1} = -\alpha \nabla_1 \left(L^1(\theta^1, \theta^2 + \perp f_2^n)\right) \approx -\alpha \left(\nabla_1 L^1 + \nabla_{12} L^1 f_2^n\right)$$
$$f_2^{n+1} = -\alpha \nabla_2 \left(L^2(\theta^1 + \perp f_1^n, \theta^2)\right) \approx -\alpha \left(\nabla_2 L^2 + \nabla_{21} L^2 f_1^n\right) \, ,$$

where $\perp$ is the stop-gradient operator (see (Balduzzi et al., 2018) for details on this operator) and $f_1^{-1} = f_2^{-1} = 0$. This can be written more succinctly as

$$\begin{pmatrix} f_1^{n+1} \\ f_2^{n+1} \end{pmatrix} = -\alpha \begin{pmatrix} \nabla_1 L^1 + \nabla_{12} L^1 f_2^n \\ \nabla_2 L^2 + \nabla_{21} L^2 f_1^n \end{pmatrix} = -\alpha\xi - \alpha H_o \begin{pmatrix} f_1^n \\ f_2^n \end{pmatrix} \, .$$

We prove by induction that

$$\begin{pmatrix} f_1^n \\ f_2^n \end{pmatrix} = -\alpha \sum_{i=0}^{n} (-\alpha H_o)^i \xi$$

for all $n \geq 0$. The base case is trivial; assume the statement holds for any fixed $n \geq 0$. Then

$$\begin{pmatrix} f_1^{n+1} \\ f_2^{n+1} \end{pmatrix} = -\alpha\xi - \alpha H_o \left(-\alpha \sum_{i=0}^{n} (-\alpha H_o)^i \xi\right) = -\alpha\xi - \alpha \sum_{i=1}^{n+1} (-\alpha H_o)^i \xi = -\alpha \sum_{i=0}^{n+1} (-\alpha H_o)^i \xi$$

as required. Finally we conclude

$$\text{LookAhead}n = \begin{pmatrix} f_1^n \\ f_2^n \end{pmatrix} = -\alpha \sum_{i=0}^{n} (-\alpha H_o)^i \xi = \text{CGD}n$$

as required. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad\square$

E. Proof of Proposition 4.5

We prove that the two pairs of linear functions

$$f_1 = f_2 = -2(x + y + 1)$$

and

$$f_1 = f_2 = -\frac{1}{2}(x + y - 2)$$

are solutions to the consistency equations in the Tandem game with $\alpha = 1$. (See below for a generalization to any $\alpha > 0$.) For the first pair of functions, we have

$$-\nabla_x \left(L^1(x, y + f_2) \right) = -\nabla_x \left((x + y + 2)^2 - 2x \right) = -2(x + y + 1) = f_1$$

for the first consistency equation and similarly

$$-\nabla_y \left(L^2(x + f_1, y) \right) = -\nabla_x \left((x + y + 2)^2 - 2y \right) = -2(x + y + 1) = f_2$$

for the second. For the second pair of functions we similarly obtain

$$-\nabla_x \left(L^1(x, y + f_2) \right) = -\nabla_x \left(\frac{1}{4}(x + y + 2)^2 - 2x \right) = -\frac{1}{2}(x + y - 2) = f_1$$

for the first consistency equation and

$$-\nabla_y \left(L^2(x + f_1, y) \right) = -\nabla_x \left(\frac{1}{4}(x + y + 2)^2 - 2y \right) = -\frac{1}{2}(x + y - 2) = f_2$$

for the second. This shows that both functions are solutions to the consistency equations. For general $\alpha > 0$, we can similarly show that $f_1 = f_2 = ax + by + c$ with

$$a = \frac{\pm\sqrt{1 + 8\alpha} - 1 - 4\alpha}{4\alpha} \quad ; \quad b = \frac{-2\alpha(1 + a)}{1 + 2\alpha(1 + a)} \quad ; \quad c = \frac{2\alpha}{1 + 2\alpha(1 + a)}$$

are two distinct solutions (depending on $\pm$) to the consistency equations in the Tandem game, noting that the denominators cannot be 0 for $\alpha > 0$ (otherwise leading to a contradiction in the expression for a). This is left to the reader, noting that the proof for $\alpha = 1$ is sufficient to establish that consistent solutions are not always unique. $\square$

F. Proof of Proposition 4.6

Recall from the proof of Proposition 4.5 that the linear functions

$$f_1 = f_2 = -2(x + y + 1)$$

are consistent solutions to the Tandem game with $\alpha = 1$. The SFPs of the Tandem game are $(x, 1 - x)$ for each $x \in \mathbb{R}$, but none of these are preserved by the consistent solutions above since

$$f_1(x, 1 - x) = f_2(x, 1 - x) = -4 \neq 0 \,.$$

We conclude that consistency does not imply preservation of SFPs.

Moreover, we prove that any (non-zero) linear solution to the consistency equations cannot preserve more than one SFP in the Tandem game, for any opponent shaping rate α. Assuming it did, we must have linear functions

$$f_1 = ax + by + c$$
$$f_2 = a'x + b'y + c'$$

satisfying

$$f_1(x, 1 - x) = 0 = f_1(x', 1 - x')$$

for some $x \neq x' \in \mathbb{R}$. Subtracting RHS from LHS we obtain $(x - x')(a - b) = 0$ hence $a = b$, which substituted again into the LHS yields $b = -c$. Applying the same method for f_2 we obtain $a' = b' = -c'$ and so f_1, f_2 take the form

$$f_1 = a(x + y - 1)$$
$$f_2 = a'(x + y - 1).$$

Note that since f_1, f_2 were assumed to be nonzero, it follows that $a, a' \neq 0$. Plugging these into the first consistency equation, we obtain

$$a(x + y - 1) = -2\alpha \left[(1 + a') \left((x + y)(1 + a') - a' \right) - 1 \right] \,.$$

Comparing x terms and constant terms yields

$$a = -2\alpha(1 + a')^2 \quad \text{and} \quad a = -2\alpha \left(1 + a' + a'^2 \right)$$

which concludes the contradiction $a' = 0$. $\square$

G. Proof of Proposition 4.7

LOLA and SOS diverge. Assume $(x_0, y_0) \neq 0$ and $\alpha > 1$. We prove the more general claim that p-LOLA diverges for any $0 \leq p \leq 1$ (where p may take a different value at each learning step), recalling that LOLA and SOS are both special cases of p-LOLA (Letcher et al., 2019b). Indeed, the p-LOLA gradient update is given by

$$\begin{pmatrix} h_1 \\ h_2 \end{pmatrix} = -\alpha(I - \alpha H_o)\xi + p\alpha^2 \chi = -\alpha \begin{pmatrix} y + \alpha x(1 + p) \\ -x + \alpha y(1 + p) \end{pmatrix}$$

and we show that each update leads to increasing distance from the origin as follows:

$$\begin{aligned}
\|(x + h_1, y + h_2)\|^2 &= x^2 - 2x\alpha(y + \alpha x(1 + p)) + \alpha^2 \left(y^2 + \alpha^2 x^2 (1 + p)^2 + 2\alpha xy(1 + p)\right) + \\
&\quad y^2 - 2y\alpha(-x + \alpha y(1 + p)) + \alpha^2 \left(x^2 + \alpha^2 y^2 (1 + p)^2 - 2\alpha xy(1 + p)\right) \\
&= (x^2 + y^2)\left(1 - \alpha^2(2p + 1) + \alpha^4(1 + p)^2\right) \\
&\geq (x^2 + y^2)\left(1 - \alpha^2 + \alpha^4\right) := \|(x, y)\|^2 \lambda
\end{aligned}$$

where the inequality follows because the final expression in p has positive derivative for $\alpha > 1$, hence minimized at $p = 0$. Now $\lambda > 1$ for any $\alpha > 1$, so we conclude by induction that

$$\|(x_n, y_n)\|^2 \geq \lambda^n \|(x_0, y_0)\|^2 \to \infty$$

as $n \to \infty$, provided $(x_0, y_0) \neq 0$, as required.

Consistent solution converges. We begin by showing that the following linear functions satisfy the consistency equations for the Hamiltonian game:

$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} = \frac{-\alpha}{1 + 2\alpha^2} \begin{pmatrix} y + 2\alpha x \\ -x + 2\alpha y \end{pmatrix}.$$

Indeed, the RHS of the first consistency equation is

$$\begin{aligned}
-\alpha \nabla_x \left(x \left(y - \alpha \frac{-x + 2\alpha y}{1 + 2\alpha^2}\right)\right) &= \frac{-\alpha}{1 + 2\alpha^2}\left(y(1 + 2\alpha^2) - \alpha(-x + 2\alpha y) + \alpha x\right) \\
&= \frac{-\alpha}{1 + 2\alpha^2}\left(y + 2\alpha x\right) = f_1
\end{aligned}$$

and similarly for the second equation.

To prove uniqueness, assume there is a second pair of linear update functions $\hat{f}_1, \hat{f}_2$ also satisfying consistency. Let $a, b, c \in \mathbb{R}$ such that $\hat{f}_1(x, y) = ax + by + c$. Note that substituting the second equation into the first yields

$$\begin{aligned}
\hat{f}_1(x, y) &= -\alpha \nabla_x \left(L^1(x, y - \alpha \nabla_y \left(L^2(x + \hat{f}_1(x, y), y)\right))\right) \\
&= -\alpha \left(y + \alpha \left(2x + \hat{f}_1(x, y) + x\nabla_x \hat{f}_1(x, y) + y\nabla_y \hat{f}_1(x, y) + xy\nabla_{xy}\hat{f}_1(x, y)\right)\right)
\end{aligned}$$

Expanding the above and substituting the equation for $\hat{f}_1$, we obtain

$$ax + by + c = -2\alpha^2 x(1 + a) - \alpha y(1 + 2\alpha b) - \alpha^2 c$$

for all $x, y \in \mathbb{R}$, which yields (by comparing coefficients)

$$a(1 + 2\alpha^2) = -2\alpha^2 \quad ; \quad b(1 + 2\alpha^2) = -\alpha \quad ; \quad c(1 + \alpha^2) = 0.$$

It follows that

$$\hat{f}_1(x, y) = \frac{-\alpha}{1 + 2\alpha^2}\left(y + 2\alpha x\right) = f_1(x, y),$$

proving the uniqueness of f_1. Since f_2 is directly determined by f_1 via the second consistency equation, this concludes the proof.

Finally we prove that this linear update leads to decreasing distance from the origin as follows:

$$\|(x + f_1, y + f_2)\|^2 = x^2 - \frac{2x\alpha}{1 + 2\alpha^2}(y + 2\alpha x) + \frac{\alpha^2}{(1 + 2\alpha^2)^2}\left(y^2 + 4\alpha^2 x^2 + 4\alpha xy\right) +$$

$$y^2 - \frac{2y\alpha}{1 + 2\alpha^2}\alpha(-x + 2\alpha y) + \frac{\alpha^2}{(1 + 2\alpha^2)^2}\left(x^2 + 4\alpha^2 y^2 - 4\alpha xy\right)$$

$$= \left(x^2 + y^2\right)\left(1 - \frac{\alpha^2(3 + 4\alpha^2)}{(1 + 2\alpha^2)^2}\right) := \|(x, y)\|^2 \lambda.$$

Notice that the derivative of λ is strictly negative in α while its limit as $\alpha \to \infty$ is 0, with value 1 at $\alpha = 0$, hence $|\lambda| = \lambda < 1$ for any $\alpha > 0$. We conclude by induction that

$$\|(x_n, y_n)\|^2 = \lambda^n \|(x_0, y_0)\|^2 \to 0$$

as $n \to \infty$, with λ decreasing (hence the speed of convergence increasing) as α increases. $\qquad\square$

H. Training Details COLA

All code was implemented using Python. The code relies on the PyTorch library for autodifferentiability (Paszke et al., 2019).

H.1. Polynomial games

For the polynomial games, COLA uses a neural network with 1 non-linear layer for both $h_1(\theta^1, \theta^2)$ and $h_2(\theta^1, \theta^2)$. The non-linearity is a ReLU function. The layer has 8 nodes. For training, we randomly sample pairs of parameters on a [-1, 1] parameter region. In general, the size of the region is a hyperparameter. We use a batch size of 8. We found that training is improved with a learning rate scheduler. For the learning rate scheduling we use a γ of 0.9. We train the neural network for 120,000 steps. To compute the consistency loss we use the squared distance measure. The optimizer used is Adam (Kingma & Ba, 2015).

H.2. Non-polynomial games

For the non-polynomial games, we deploy a neural network with 3 non-linear layers using Tanh activation functions. Each layer has 16 nodes. For this type of game, the parameter region is set to [-7, 7], because the parameters will be squished into probability space, allowing us to explore the full probability space. During training, we used a batch size of 64. The optimizer used is Adam (Kingma & Ba, 2015).

I. Further experimental results

I.1. Tandem game

In this section, we will provide more empirical results on the Tandem game. First, in Figure 4, we display a look-ahead regime where SOS, LOLA and HOLA8 diverge in training, whereas COLA and CGD do not. Second, in Figure 5 we compare the gradient fields of HOLA4 and COLA at a low and high look-ahead rate of 0.1 and 1.0 respectively. The updates are shown on the parameter region of interesting Θ for the Tandem game, which is [-1,1]. Figure 5a and 5c show that the solutions are very similar at low look-ahead rates but become dissimilar at high look-ahead rates, shown in Figure 5b and 5d. This also confirms our quantitative observation in Table 1b.
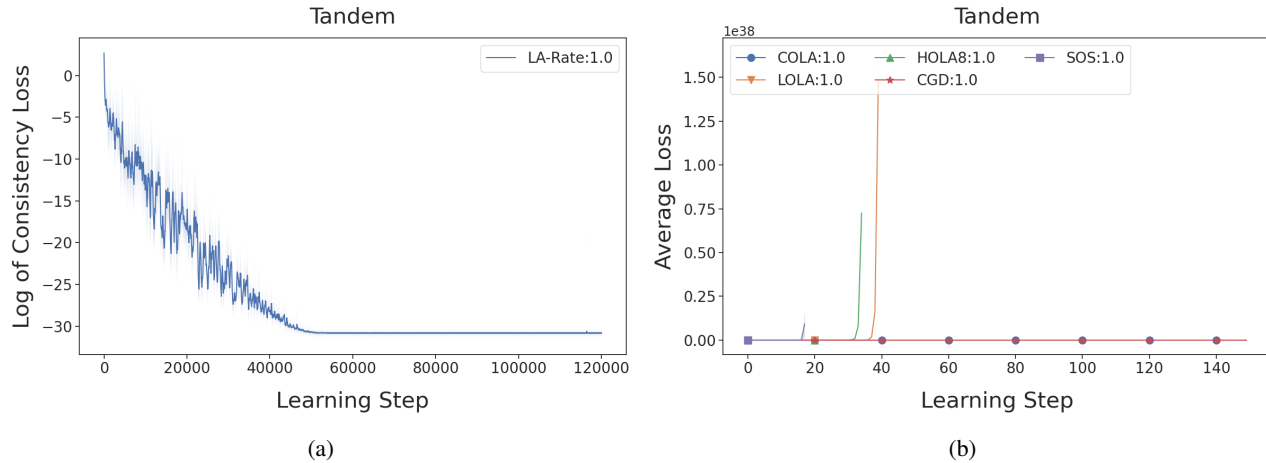

(a)   (b)

Figure 4. (a): Consistency loss of COLA at a look-ahead rate of 1.0. (b): Solutions on the Tandem game by COLA, LOLA, HOLA8, CGD, and SOS with a look-ahead rate of 1.0. The standard deviation for the initialization of parameters used here is 0.1, which is standard in the literature (Letcher et al., 2019b).
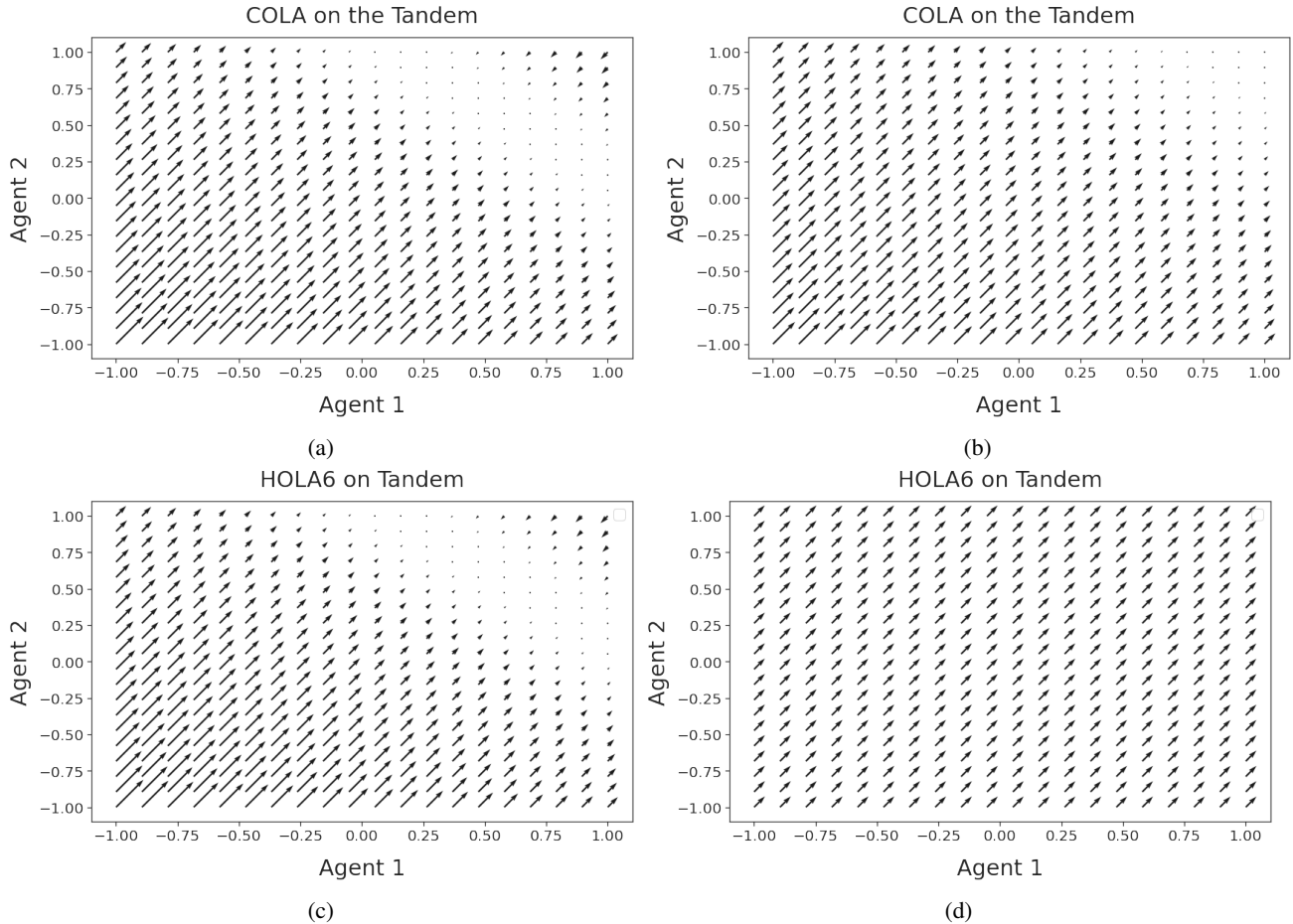
Figure 5. Gradients field of the Tandem game at two different look-ahead rates, 0.1 and 1.0.

I.2. Balduzzi and Hamiltonian game

The Hamiltonian game was originally introduced in (Balduzzi et al., 2018) as a minimal example of Hamiltonian dynamics. Recall that its loss function is

$$L^1(x, y) = xy \quad \text{and} \quad L^2(x, y) = -xy \tag{34}$$

The Balduzzi game was introduced to investigate the behaviour of differentiable game algorithms when a weak attractor is coupled with strong rotational forces in the Hamiltonian dynamics (Balduzzi et al., 2018), captured by the losses

$$L^1(x, y) = \frac{1}{2}x^2 + 10xy \quad \text{and} \quad L^2(x, y) = \frac{1}{2}y^2 - 10xy \tag{35}$$

Results for both games are displayed in Figures 6 and 7. COLA at high look-ahead rates converges considerably faster than the other methods on both games. This supports Proposition 4.7 empirically. In Table 4a, 4b and 5a we find similar consistency loss behaviour as we do for Tandem. Note however, that the look-ahead thresholds are at significantly different levels. On the Balduzzi game, the threshold is much lower around 0.02, whereas for the Hamiltonian game it is between 0.1 and 0.4.
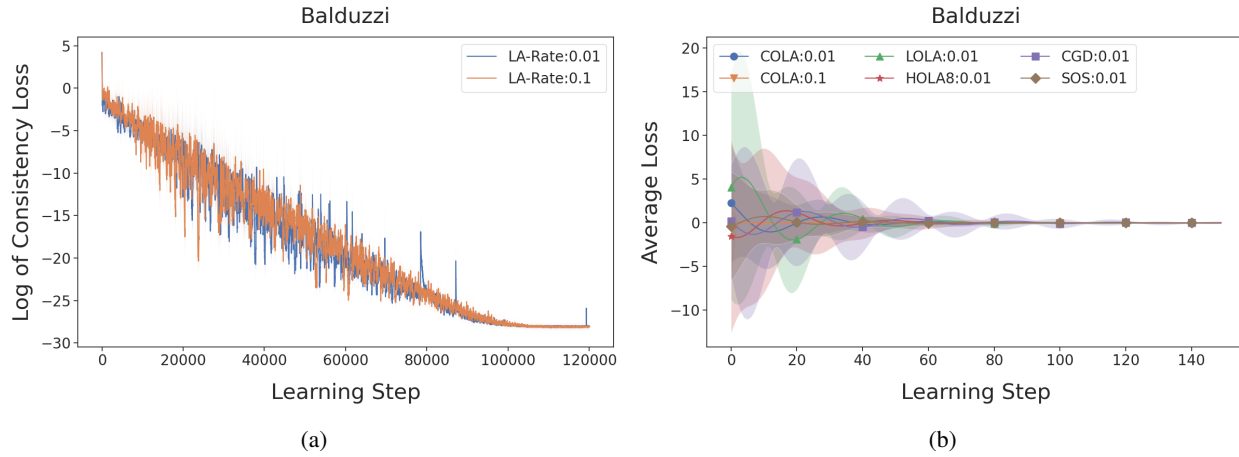
(a)

(b)

Figure 6. (a): Consistency losses of COLA at different look-ahead rates. (b): Solutions on the Balduzzi game by COLA, LOLA, HOLA8, CGD, and SOS with a look-ahead rate of 0.01 (and 0.1 additionally for COLA). The standard deviation for the initialization of parameters used here is 1.0
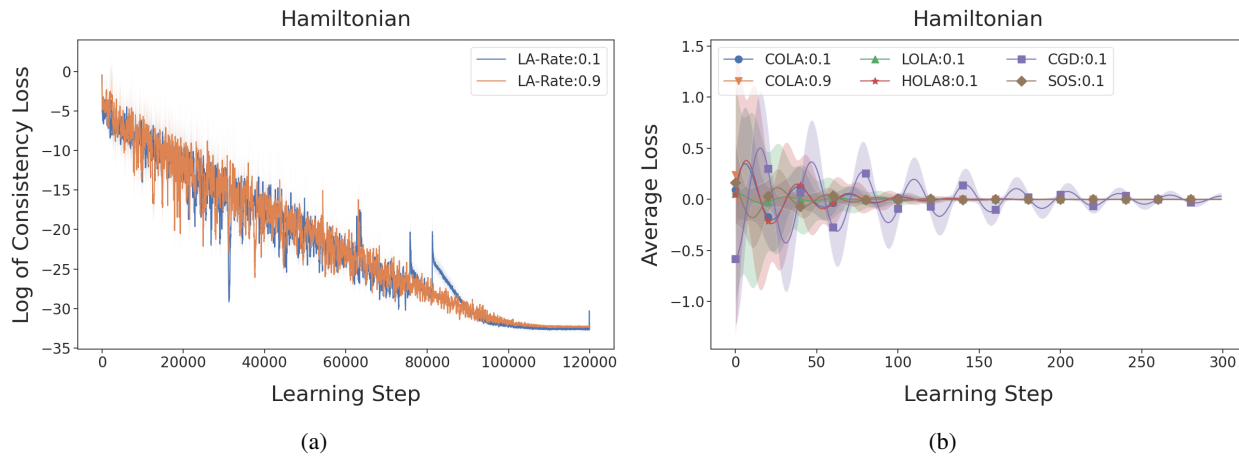

(a)

(b)

Figure 7. (a): Consistency losses of COLA at different look-ahead rates. (b): Solutions on the Hamiltonian game by COLA, LOLA, HOLA8, CGD, and SOS with a look-ahead rate of 0.1 (and 0.9 additionally for COLA). The standard deviation for the initialization of parameters used here is 1.0

I.3. Matching Pennies

Matching Pennies (MP) is a single-shot, zero-sum game, where two players, A and B, each flip a biased coin (Lee & K, 1967). Player A wins if the outcomes of both flips are the same and player B wins if they are different.

The visual difference between the updates found by COLA and HOLA4 is shown in Figure 8. Whereas they are very similar at a relatively low look-ahead rate, at a high look-ahead rate HOLA4's gradient field shows signs of high variance, especially around the origin, whereas COLA's gradient field appears to learn a robust update function. Note that HOLA4's update function results in the learning behaviour shown in Figure 2b.

Table 4. On the Hamiltonian game: (a) Log of the squared consistency loss. (b) Cosine similarity between COLA and LOLA, HOLA3, and HOLA6 over different look-ahead rates. The values represent the mean of a 1,000 samples, uniformly sampled from the parameter space Θ. The error bars represent one standard deviation and capture the variance over 10 different COLA training runs.

(a)

α	LOLA	HOLA3	HOLA6	COLA
0.9	6.99	4.63	13.68	1e-14$\pm$2e-15
0.5	12.67	13.77	13.00	1e-14$\pm$2e-15
0.4	5.78	7.14	6.38	1e-14$\pm$2e-15
0.1	0.08	0.01	2e-6	7e-15$\pm$1e-15
0.05	2e-5	4e-10	3e-15	1e-14$\pm$2e-14

(b)

α	LOLA	HOLA3	HOLA6
0.9	1.00$\pm$0.00	-1.00$\pm$0.00	0.50$\pm$0.00
0.5	1.00$\pm$0.00	1.00$\pm$0.00	1.00$\pm$0.00
0.4	1.00$\pm$0.00	1.00$\pm$0.00	1.00$\pm$0.00
0.1	1.00$\pm$0.00	1.00$\pm$0.00	1.00$\pm$0.00
0.05	1.00$\pm$0.00	1.00$\pm$0.00	1.00$\pm$0.00

Table 5. On the Balduzzi game: (a) Log of the squared consistency loss. (b) Cosine similarity between COLA and LOLA, HOLA3, and HOLA6 over different look-ahead rates. The values represent the mean of a 1,000 samples, uniformly sampled from the parameter space Θ. The error bars represent one standard deviation and capture the variance over 10 different COLA training runs.

(a)

α	LOLA	HOLA3	HOLA6	COLA
0.9	2e+6	5e+10	4e+17	5e-13$\pm$1e-13
0.1	3e+2	1.e+3	2e+4	7e-13$\pm$1e-13
0.05	2e+1	4.01	1.03	7e-13$\pm$9e-14
0.03	2.16	0.07	0.01	8e-13$\pm$1e-13
0.01	0.03	1e-5	2e-10	7e-13$\pm$1e-13

(b)

α	LOLA	HOLA3	HOLA6
0.9	1.00$\pm$0.00	-1.00$\pm$0.00	0.01$\pm$4e-9
0.1	1.00$\pm$0.00	1.00$\pm$0.00	0.40$\pm$1e-8
0.05	1.00$\pm$0.00	1.00$\pm$0.00	1.00$\pm$0.00
0.03	1.00$\pm$0.00	1.00$\pm$0.00	1.00$\pm$0.00
0.01	1.00$\pm$0.00	1.00$\pm$0.00	1.00$\pm$0.00

Table 6. Payoff Matrix for the Matching Pennies game.

	Head	Tail
Head	(+1, -1)	(-1, +1)
Tail	(-1, +1)	(+1, -1)

Table 7. Cosine similarities over multiple COLA training runs on the MP and Tandem game for different look-ahead rates.

Game@LR	Cosine Sim
MP@10	0.97 $\pm$ 0.01
MP@0.5	0.99 $\pm$ 0.01
Tandem@0.1	1.00 $\pm$ 0.00
Tandem@1.0	0.98 $\pm$ 0.01

I.4. Ultimatum game

Here we provide more empirical results on the Ultimatum game. First, in Table 8a and 8b we display the consistency losses of COLA and HOLA at different look-ahead rates. In comparison to the polynomial games, here we observe that COLA's consistency losses are not as low as HOLAn's losses at low look-ahead rates. Nonetheless, they are low enought to constitute a consistent solution. Moreover, the COLA and HOLAn solutions are very similar according to the cosine similarity score. Qualitatively, we compare the updates in Figure 9. Similarly to the MP game, we observe that HOLA4's update shows higher variance around the origin that COLA's update. This variance is reflected in HOLA4's learning behaviour shown in Figure 10d, where HOLA4 do not converge to the fair solution consistently.
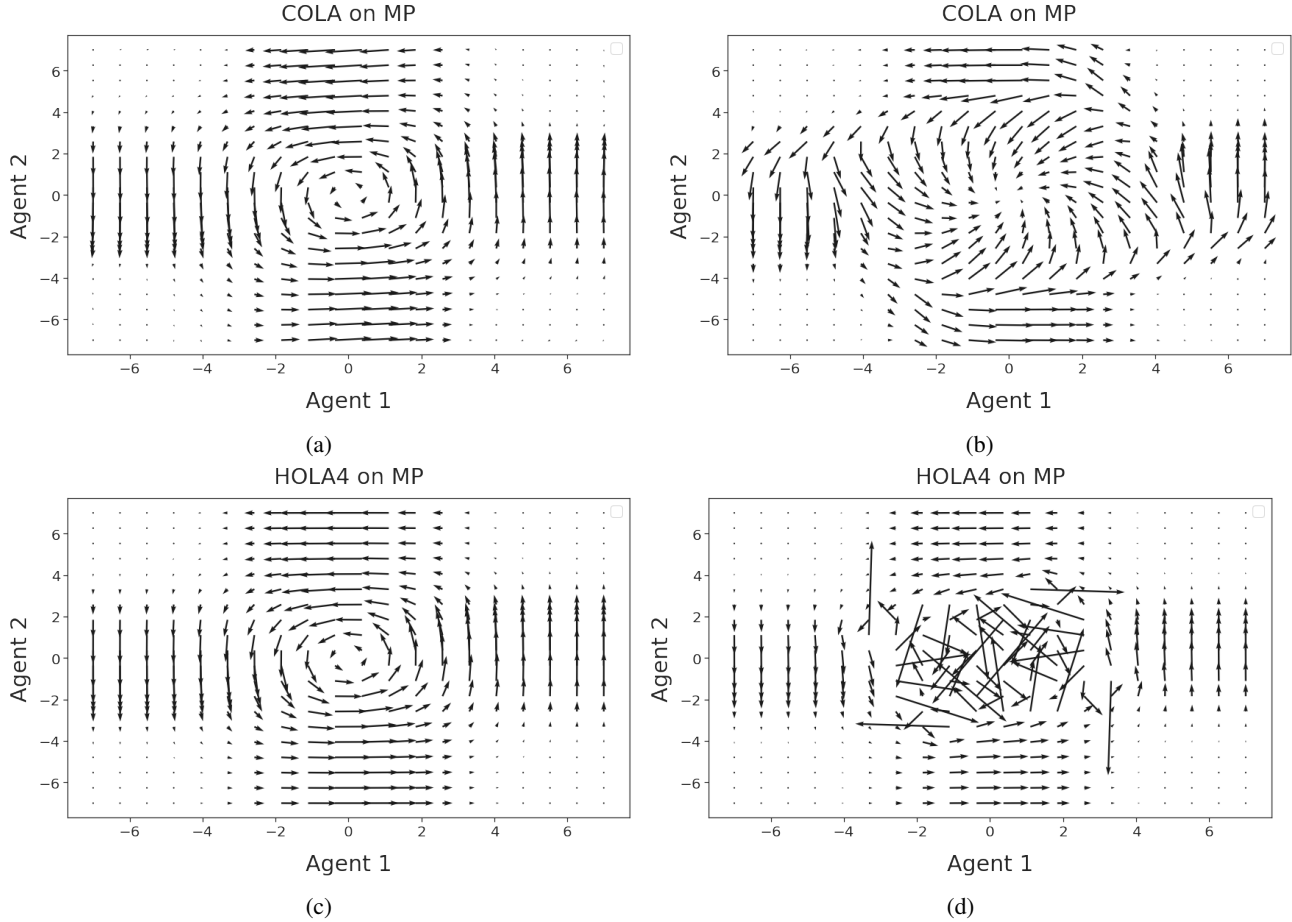
(a)

(b)

(c)

(d)

Figure 8. Gradients field of the MP game at two different look-ahead rates: 0.5 (RHS) and 10 (LHS). COLA is on the upper row, HOLA4 is on the lower row.

Table 8. On the Ultimatum game: Over multiple look-ahead rates we compare (a) the consistency losses and (b) the cosine similarity between COLA and LOLA, HOLA2, and HOLA4. The values represent the mean of a 1,000 samples, uniformly sampled from the parameter space Θ. The error bars represent one standard deviation and capture the variance over 10 different COLA training runs.

<table>
<tr><td colspan="5" align="center">(a)</td><td colspan="4" align="center">(b)</td></tr>
<tr><td>α</td><td>LOLA</td><td>HOLA2</td><td>HOLA4</td><td>COLA</td><td>α</td><td>LOLA</td><td>HOLA2</td><td>HOLA4</td></tr>
<tr><td>1.1</td><td>2e-3</td><td>5e-3</td><td>0.01</td><td>4e-4±5e-5</td><td>1.1</td><td>0.96±0.01</td><td>0.96±0.01</td><td>0.95±0.01</td></tr>
<tr><td>0.7</td><td>4e-4</td><td>2e-4</td><td>2e-4</td><td>7e-5±1e-5</td><td>0.7</td><td>0.98±0.01</td><td>0.98±0.01</td><td>0.98±0.01</td></tr>
<tr><td>0.3</td><td>3e-5</td><td>2e-7</td><td>5e-8</td><td>3e-6±2e-6</td><td>0.3</td><td>0.99±0.01</td><td>0.99±0.01</td><td>0.99±0.01</td></tr>
<tr><td>0.1</td><td>2e-7</td><td>1e-11</td><td>6e-13</td><td>4e-6±4e-6</td><td>0.1</td><td>0.99±0.01</td><td>0.99±0.01</td><td>0.99±0.01</td></tr>
<tr><td>0.001</td><td>3e-15</td><td>4e-17</td><td>9e-17</td><td>4e-6±3e-6</td><td>0.001</td><td>0.99±0.01</td><td>0.99±0.01</td><td>0.99±0.01</td></tr>
</table>

I.5. Chicken game

In the Chicken game, an agent can either choose to yield to avoid a catastrophic payoff but face a small punishment if they are the only agent to yield. Imagine a game where two agents drive towards each other in their cars. If both never swerve, they frontally crash into each other, an obviously catastrophic outcome. If any of the agents "chicken out", e.g. swerve, they do not crash but receive a small punishment for having chickened out. At the same time, the other agent is being rewarded for staying on track, as quantified in Table 9.

Next we report the consistency losses on the Chicken game in Table 10a. We identify a look-ahead rate threshold where HOLAn's consistency loss becomes increasingly bigger with increasing order. For the Chicken game, this threshold is fairly
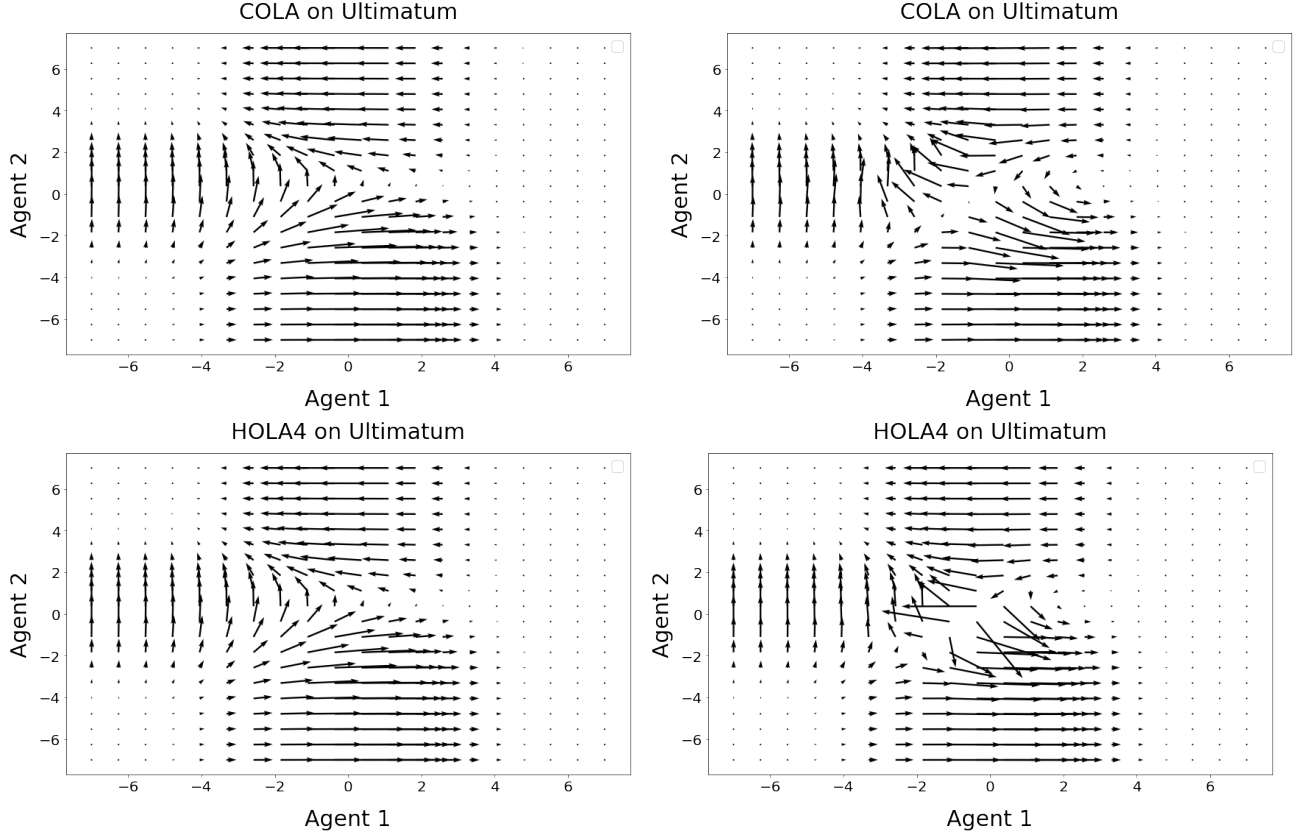
Figure 9. Gradients field of the ultimatum game at two different look-ahead rates: 0.2 (RHS) and 1.1 (LHS). COLA is on the upper row, HOLA4 is on the lower row.

Table 9. Payoff Matrix for the Chicken game.

	C (swerve)	D (straight)
C (swerve)	0, 0	-1, +1
D (straight)	+1 ,-1	-100, -100

low, between 0.01 and 0.05. Interestingly, we note that around a look-ahead rate of 0.05 and 0.1 it becomes harder to find a consistent solution for COLA.

Table 10. On the Chicken game: Over multiple look-ahead rates we compare (a) the consistency losses and (b) the cosine similarity between COLA and LOLA, HOLA2, and HOLA4. The values represent the mean of a 1,000 samples, uniformly sampled from the parameter space Θ. The error bars represent one standard deviation and capture the variance over 10 different COLA training runs.

(a)

α	LOLA	HOLA2	HOLA4	SOS	CGD	COLA
1.0	2429	3892	46637	1494	1677	0.01±0.01
0.5	643	484	4320	475	2330	0.03±0.01
0.1	11.99	7.69	73.28	2.73	8.46	0.70±0.10
0.05	0.84	0.17	0.47	0.37	1.31	0.06±0.01
0.01	9e-4	3e-6	6e-9	2e-4	0.04	5e-4±3e-4

(b)

α	LOLA	HOLA2	HOLA4
1.0	0.39±0.03	0.57±0.02	0.57±0.03
0.5	0.54±0.04	0.62±0.03	0.64±0.04
0.1	0.88±0.03	0.88±0.03	0.86±0.03
0.05	0.93±0.03	0.93±0.03	0.93±0.03
0.01	0.96±0.02	0.96±0.02	0.96±0.01

We also perform a qualitative comparison on the gradient fields of COLA and HOLA4 on the Chicken game (see Figure 11). The difference at high look-ahead rates is pronounced, as HOLA4 shows high variance around the origin. Nonetheless, the gradient field leads to swerving in the actual game (see Figure 12d, which is a preferable outcome. Moreover, HOLA4
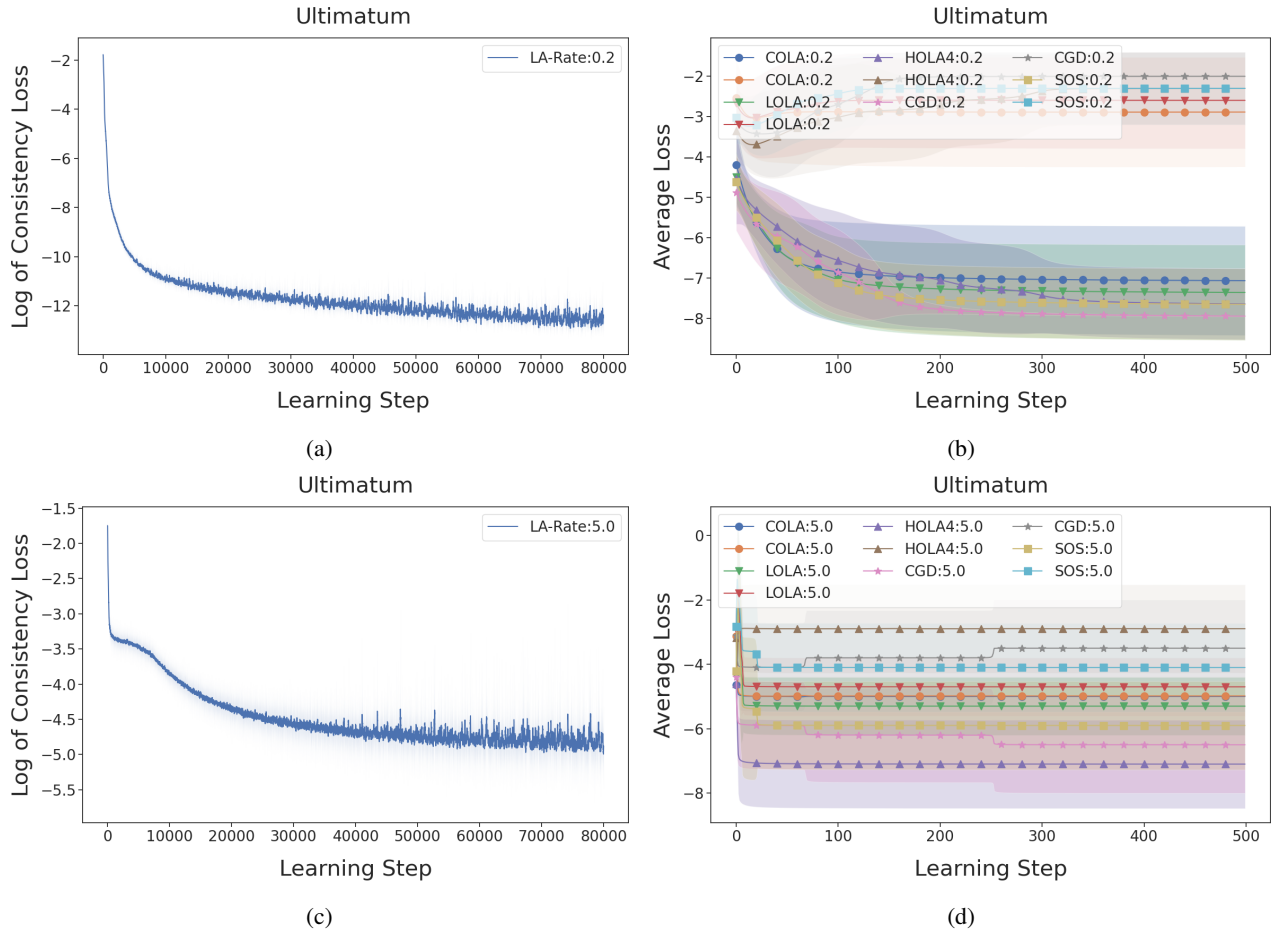
Figure 10. (a) and (c): Consistency losses of COLA at different look-ahead rates. (b) and (d): Solutions on the Ultimatum game by COLA, LOLA, HOLA4, CGD, and SOS with a look-ahead rate of 0.2 and 5.0. The standard deviation for the initialization of parameters used here is 1.0.

appears to converge to Swerving consistently despite the chaotic gradient field, showcasing that the behaviour in the game is not necessarily correlated with the qualitative analysis of the gradient fields.
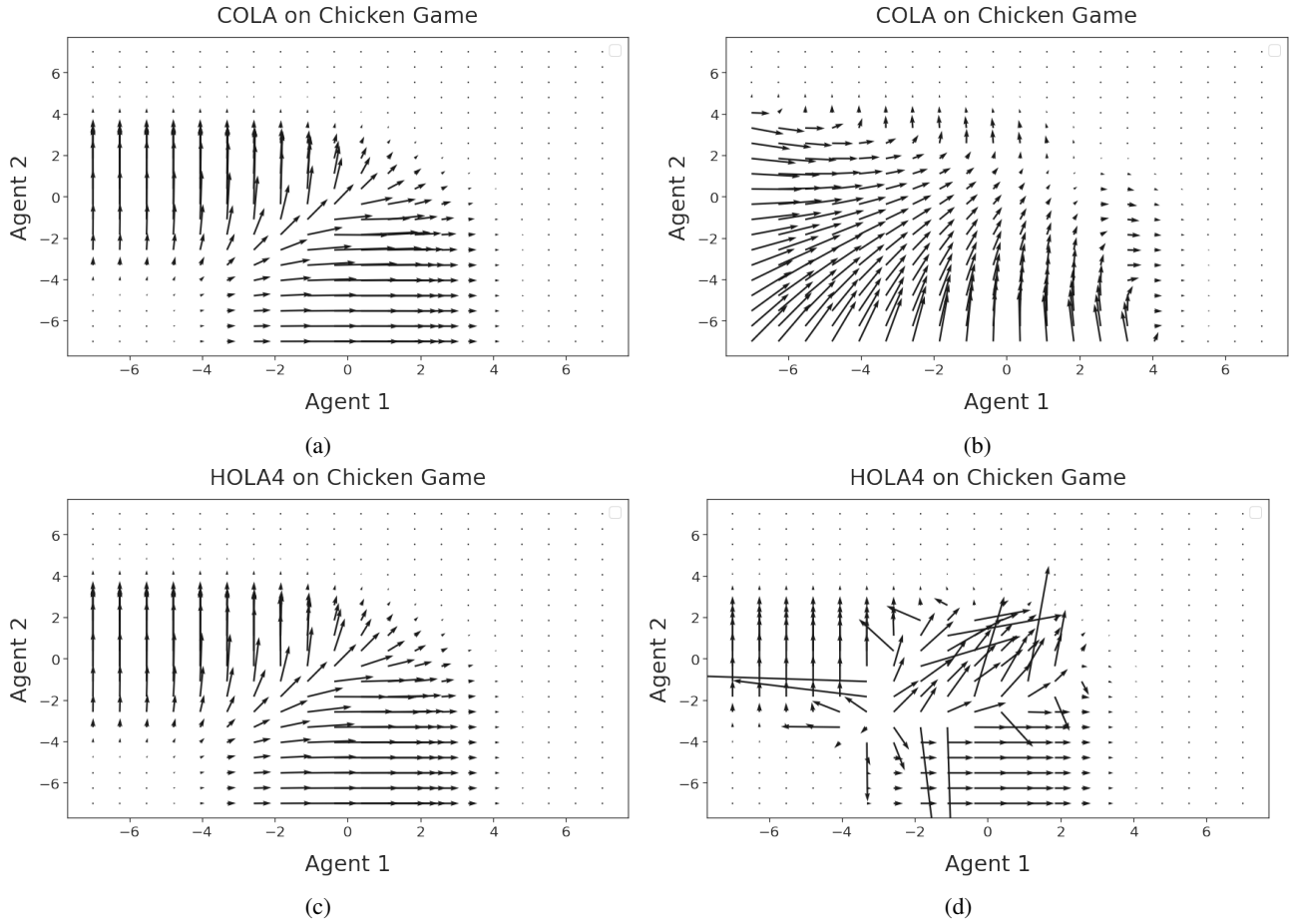
(a)

(b)

(c)

(d)

Figure 11. Gradients field of the Chicken game for COLA and HOLA4 at two different look-ahead rates, 0.01 (LHS) and 1.0. (RHS)

I.6. IPD

As a baseline comparison, we show the performance of different state-of-the-art algorithms in Figure 13. CGD does not recover the tit-for-tat policy whereas exact LOLA, Taylor LOLA and SOS do.

Table 11. Payoff Matrix for the IPD game.

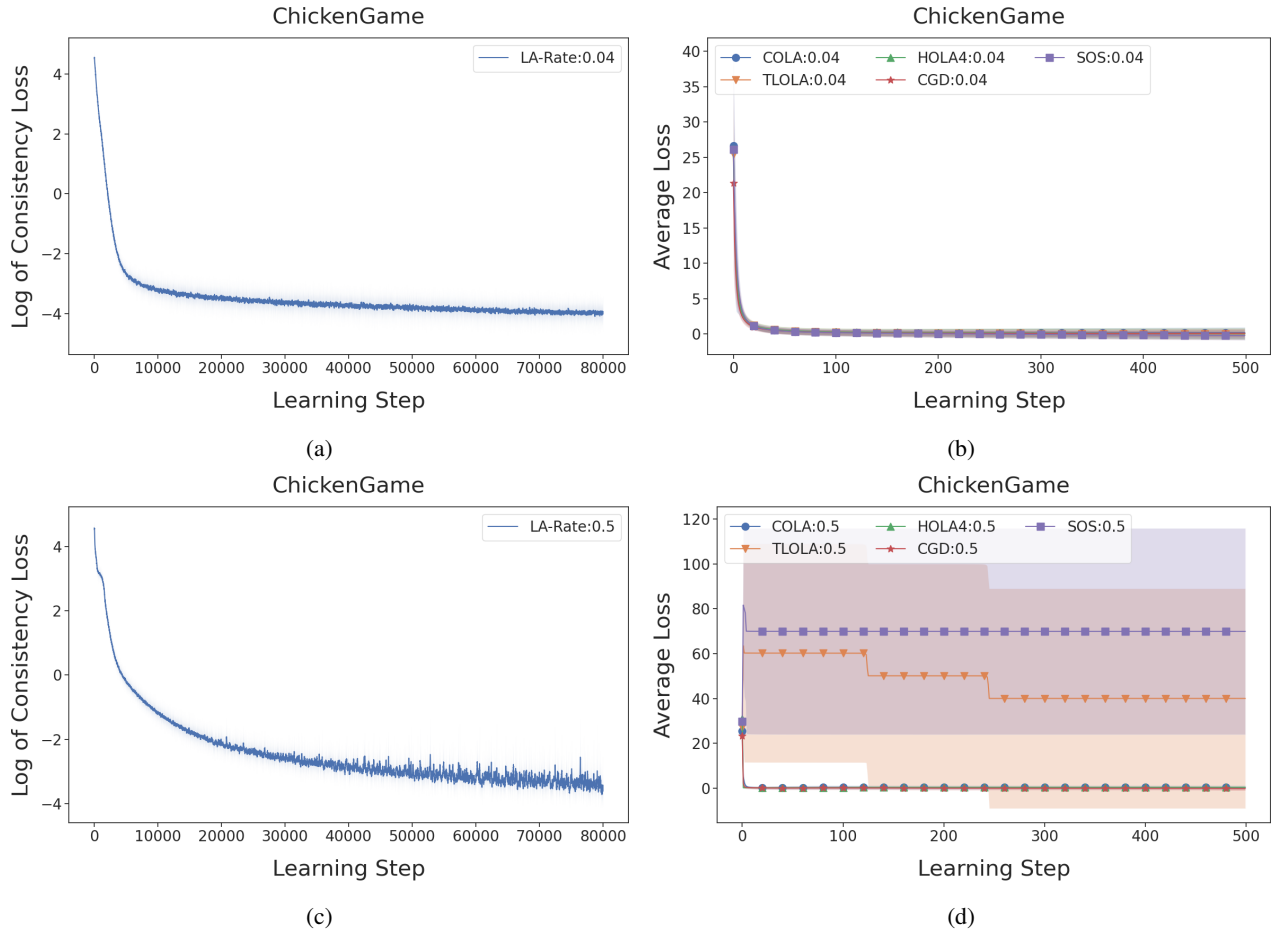|	C	D
C	(-1, -1)	(0, -3)
D	(0, -3)	(-2, -2)

(a)

(b)

(c)

(d)

Figure 12. (a) and (c): Consistency losses of COLA at different look-ahead rates. (b) and (d): Solutions on the Ultimatum game by COLA, TLOLA, HOLA4, CGD, and SOS with a look-ahead rate of 0.04 and 0.5. We used a standard deviation of 1.0 to initialize the parameters.
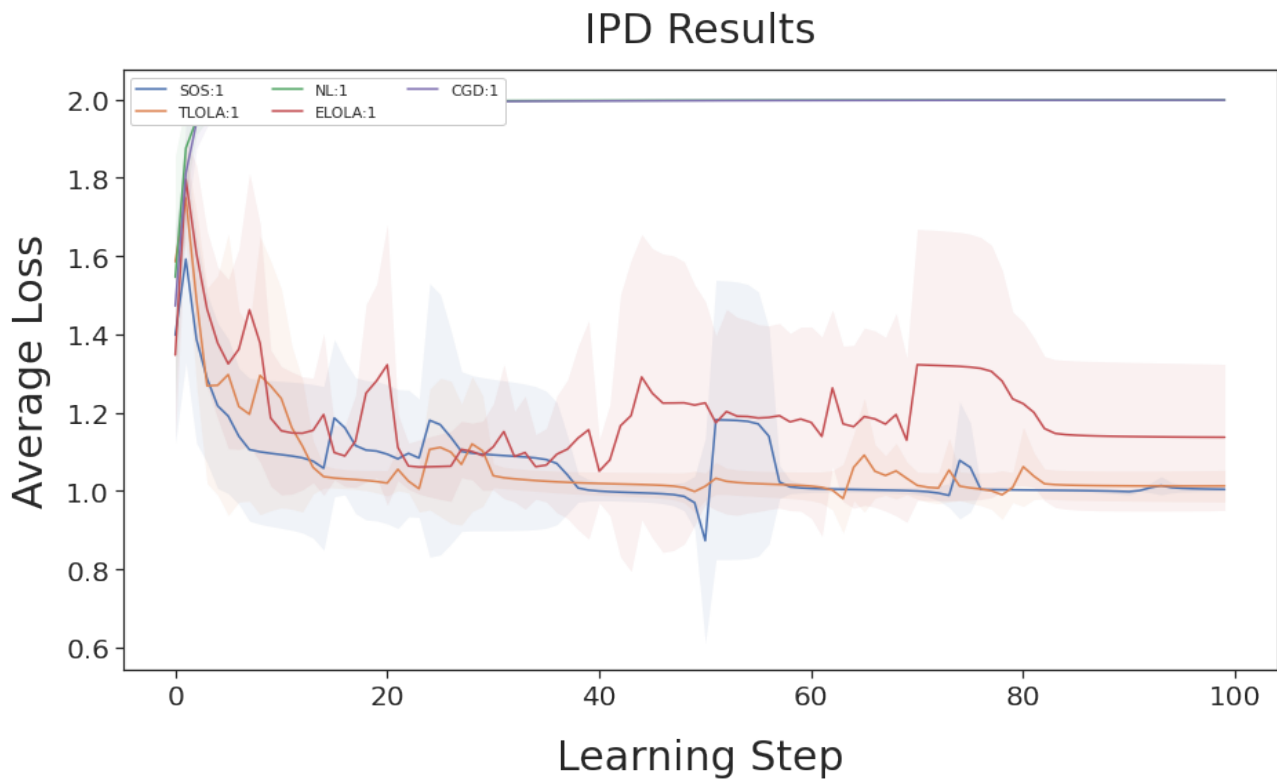
Figure 13. CGD, SOS, Taylor LOLA (TLOLA), Exact LOLA (ELOLA) and Naive Learning (NL) on the IPD at a look-ahead rate of 1.0. We used a standard deviation of 1.0 to initialize the parameters.